UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Estudio de la composición de bandas sonoras para cine mediante inteligencia artificial

Autor: Alberto Sánchez Abad
Tutor: Juan Jesús Roldán Gómez

junio 2021

Alberto Sánchez Abad
Estudio de la composición de bandas sonoras para cine mediante inteligencia artificial

Alberto Sánchez Abad
junio 2021

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

A todas las personas que me han apoyado siempre

Elige un trabajo que te guste y no tendrás que trabajar ni un día de tu vida.

Confuncio

Agradecimientos

En primer lugar me gustaría agradecer a mi tutor Juan Jesús Roldán Gómez, un grandísimo profesional, por haberme guiado durante todo este trabajo y por haber estado siempre atento a lo que necesitaba.

También me gustaría agradecer tanto a mis amigos del pueblo donde he crecido, que siempre han estado ahí, como al grupo de *Si quieres te programo* que me han acompañado durante toda esta etapa.

No puedo olvidar en este agradecimiento a mi amigo de toda la vida, Fernando, por su gran ayuda con las partes musicales más complicadas y a Emilio, el director de la banda de música de Las Navas del Marqués donde colaboro, por la distribución entre diferentes conservatorios del cuestionario elaborado.

Por último, gracias infinitas a mi familia, en especial a mis padres, a mi hermana, a mi primo Nacho y a mi novia por todo el apoyo que me han dado y la ilusión que me han mostrado con este trabajo.

Resumen

En este trabajo se realiza un estudio de la composición de bandas sonoras para cine utilizando inteligencia artificial, lo que significa que se investiga la posibilidad de crear música totalmente nueva y melódica utilizando redes neuronales.

El primer objetivo de este trabajo es conseguir generar melodías con sentido que puedan servir como ayuda en la composición de bandas sonoras. El segundo objetivo, que está estrechamente relacionado con el primero, es investigar sobre la posibilidad de clasificar las bandas sonoras según el género cinematográfico de las películas a las que pertenecen. Por último, el objetivo final, consiste en combinar los dos anteriores, es decir, intentar generar melodías de bandas sonoras, seleccionando el género cinematográfico al que se quiere que pertenezcan.

Para conseguir estos objetivos, el desarrollo de esta investigación ha constado principalmente de tres pasos. Primero se ha realizado la extracción de datos, donde se han llevado a cabo 3 estudios distintos, y para lo cual se ha hecho uso principalmente de la técnica del *Web Scraping*. Posteriormente, se ha efectuado la generación de las bandas sonoras, lo cual ha sido realizado a partir de redes neuronales recurrentes, más concretamente, de redes *Long-Short Term Memory*, ya que estas poseen una especie de memoria y dan lugar a poder conservar una línea melódica en toda la composición. Por último, se ha llevado a cabo la parte de clasificación de bandas sonoras por el género cinematográfico de las películas donde aparecen, en la cual se utilizan redes neuronales convolucionales, lo que significa que la música es convertida a imágenes antes de ser tratada por ellas.

A partir de los archivos conseguidos en la extracción de datos (alrededor de 60.000 MP3 y 565 MIDIs), se han podido generar melodías monofónicas de 12 géneros cinematográficos distintos utilizando redes neuronales recurrentes. Además, con los archivos MP3, se ha podido llevar a cabo la investigación sobre la clasificación de bandas sonoras por su género utilizando redes neuronales convolucionales. Por último, se ha hecho un estudio con personas que justifica que hay una cierta confusión a la hora de elegir el género cinematográfico simplemente escuchando las bandas sonoras. También, en este estudio, se ha determinado que las personas no son capaces de distinguir claramente entre las composiciones artificiales generadas en este trabajo y melodías de bandas sonoras reales, lo cual quiere decir que la música generada podría pasar por una pieza compuesta por un humano.

Palabras clave

Bandas sonoras, inteligencia artificial, redes neuronales, web scraping, géneros cinematográficos

Abstract

This paper describes a study about the composition of movie soundtracks using artificial intelligence, thus it explores the possibility of creating new and melodic music using neural networks.

The first goal of this project is to generate meaningful melodies which can be used as help in soundtrack composition. The second target, which is strongly related to the first one, is to investigate about the possibility of classifying soundtracks by their movie genres. Lastly, the final goal consists of combining the two above, which means to try to generate soundtrack melodies selecting the desired movie genre.

To achieve these goals, the development of this research has been composed of 3 steps. Firstly, data has been extracted and this has been performed in three different studies. These studies have been carried out with web scraping. Afterwards, the generation of soundtracks has been executed using recurrent neural networks, more specifically *Long-Short Term Memory*, which have memory and can preserve a melodic line in the composition. Finally, convolutional neural networks have been used to try to classify soundtracks by movie genre. To do this, music has been converted to pictures.

From obtained files in data extraction (around 60,000 MP3 and 565 MIDIs), monophonic melodies of 12 different movies genres could be generated using recurrent neural networks. Besides, the investigation about classifying soundtracks by movie genre could be carried out with those MP3 files. Lastly, a study with people has been performed. This study justifies that people have problems to choose the movie genre of a soundtrack. Also, this survey shows that people are not able to distinguish artificial compositions and melodies extracted from real soundtracks, which means that the new music could come across as a musical piece composed by a person.

Keywords

Movie soundtracks, artificial intelligence, neural networks, web scrapping, movie genres

Índice

Listas

Lista de figuras

1

Introducción

1.1. Motivación

En el ámbito de la investigación, la combinación de lo innovador con lo cotidiano es capaz de hacer un escenario realmente atractivo de estudiar. Precisamente, de esta unión nace la motivación de este trabajo. La Inteligencia Artificial (IA) , la parte innovadora, se define como una simulación de la inteligencia humana por parte de las máquinas. Mientras la música, la parte cotidiana, es algo tan común hoy en día que está presente en cualquier momento de la vida. La música nace de la composición musical, la cual se define como la práctica de combinar elementos musicales y sus partes para llegar a producir una pieza musical.

Hasta hace poco, era impensable que una máquina como un ordenador fuese capaz de producir música con sentido. Es lógico pensar que cualquier pieza musical está compuesta por un compositor de carne y hueso, el cual ha tenido que utilizar su propia inspiración para componerla. En este trabajo, se ha investigado la posibilidad de que un "programa" sea capaz de crear música nueva y con sentido. Estas composiciones se podrían destinar directamente al público o podrían servir de inspiración a compositores en el proceso de creación de una obra. Además, esta herramienta podría ayudar a producir una mayor variedad de canciones originales sin copyright, para que estas se pudiesen distribuir libremente y ser usadas donde se quisiese.

Uno de los campos donde esta herramienta podría ser muy útil es en el cine, uno de los grandes entretenimientos desde su invención, el cual está repleto de piezas musicales llamadas Bandas Sonoras (BSO) . Estas expresan escenas determinadas o representan una película musicalmente. Sabiendo esto y que la música se puede clasificar según los diferentes géneros musicales tales como pop, rock, jazz, clásica, blues, soul... ¿sería posible clasificar las BSO en función del género cinematográfico de la película en la que aparecen?

En este trabajo, se intenta mostrar cómo la inteligencia artificial puede llegar incluso a crear arte, en este caso en concreto, a crear algo tan sorprendente como una melodía de una banda sonora de una película.

1.2. Objetivos

El objetivo principal de este trabajo es el de estudiar y explorar la viabilidad de componer BSO a partir de IA . Para ello, se generarán distintas melodías monofónicas de forma artificial y se compararán con otras melodías monofónicas extraídas de películas. Sin embargo, antes de generar y analizar estas composiciones, es necesario entender cómo la IA , a partir de redes neuronales recurrentes, es capaz de entrenar un modelo para que este sea capaz de producir estas melodías monofónicas y, además, tengan sentido. Para conseguir este objetivo, hay que realizar un estudio sobre las distintas tecnologías disponibles y elegir la que más se adecue a nuestras pretensiones.

El segundo objetivo de este trabajo es determinar hasta qué punto es viable diferenciar las bandas sonoras en cuanto al género cinematográfico de las películas en las que aparecen. Esto se intentará probar tanto utilizando modelos de IA como contando con la participación de personas.

Por último, el objetivo final del presente Trabajo de Fin de Grado (TFG) es mezclar las anteriores premisas. Esto supone que se van a intentar generar melodías monofónicas de BSO de forma artificial, de modo que antes de crear una, se pueda decidir de qué género cinematográfico se quiere que sea la composición.

1.3. Organización de la memoria

La memoria se ha organizado de la siguiente forma:

1. Introducción: Se describe la motivación que ha dado lugar a este TFG , los objetivos que se quieren conseguir y cómo se va a estructurar esta memoria.

2. Estado del arte: Se describe todo el entorno científico necesario para entender cómo es posible llegar a los resultados presentados. Además, se estudian varias investigaciones similares a este trabajo y se comparan los temas que se abordaron en estos respecto a los que se tratan en el presente documento.

3. Desarrollo: Se describen las iteraciones que se han llevado a cabo en el proyecto de manera detallada, explicando cómo se realizó cada paso de la investigación y detallando las tecnologías utilizadas.

4. Experimentos y resultados: Se explican las pruebas llevadas a cabo tanto de manera artificial como contando con la participación de personas. Los resultados de estas pruebas son mostrados analíticamente y visualmente en forma de gráficos.

5. Conclusiones y trabajo futuro: Se extraen las conclusiones basadas en los resultados obtenidos, así como el posible trabajo futuro relacionado con este trabajo.

2

Estado del arte

En esta sección se describen los principales campos que se van a abordar en esta investigación: La IA , la música y el Web Scraping (WS) . Además, al final se realiza una investigación sobre estudios similares a los llevados a cabo en este documento y se comparan los temas abordados en cada uno.

2.1. Inteligencia Artificial

El concepto de IA fue creado tiempo atrás por el sumo interés del ser humano en dotar de inteligencia a los instrumentos creados por él mismo. Al principio sólo se trataba de mitos y leyendas, como Frankestein, obra escrita en el siglo XIX. Incluso antes de esto, en el siglo XVII, René Descartes ya se preguntaba si las máquinas podrían llegar a pensar en un futuro. [1]

Para explicar qué es la IA se va a utilizar la definición proporcionada por Lasse Rouhiainen, un experto en el tema: [2] *Es la habilidad de los ordenadores para hacer actividades que normalmente requieren inteligencia humana. Pero, para brindar una definición más detallada, podríamos decir que la IA es la capacidad de las máquinas para usar algoritmos, aprender de los datos y utilizar lo aprendido en la toma de decisiones tal y como lo haría un ser humano.*

2.1.1. Aprendizaje Automático

El Aprendizaje Automático o también conocido como Machine Learning (ML) es un campo de la IA que consiste en hacer posible que una máquina aprenda por sí sola a partir del análisis e interpretación de patrones y datos.

Esta tecnología necesita de un gran volumen de datos para conseguir una mayor inteligencia en la máquina. Esto se debe a que los algoritmos que llevan a cabo el Aprendizaje Automático necesitan encontrar patrones en esos datos para proporcionarla capacidad de decidir por sí misma.

En la figura 2.1 se puede observar un proceso de ML en el que, introduciendo unos datos de entrada a la máquina (Ejemplo), se ejecuta el algoritmo elegido para que un modelo sea capaz de clasificar u optimizar basándose en distintos patrones de los datos de entrada. Por tanto, cuando llega un caso nuevo a la máquina, esta es capaz de clasificar u optimizar ese caso devolviéndolo como respuesta.

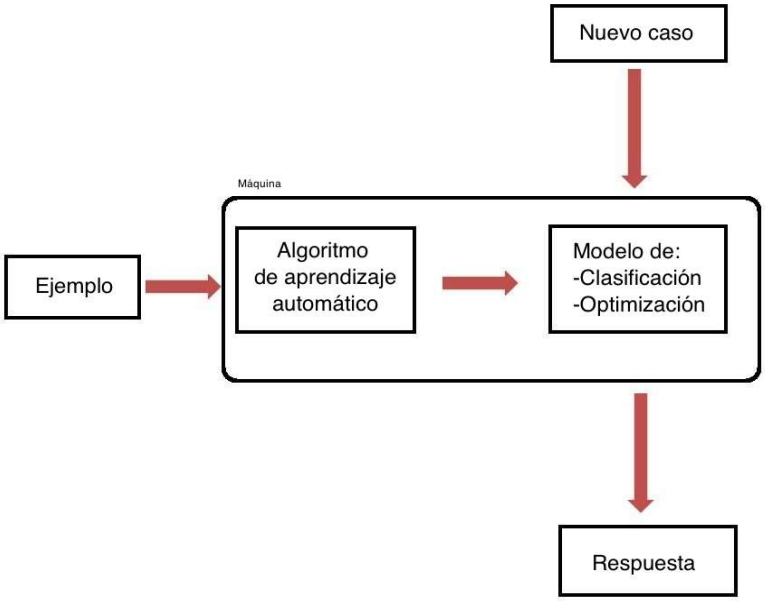


Figura 2.1: Esquema del proceso de Aprendizaje Automático [3]

2.1.2. Tipos de Aprendizaje Automático

El ML puede ser supervisado, no supervisado o por refuerzo como se indica en la figura 2.2, en la cual se muestran algunos ejemplos de uso para cada uno de los tipos:

1. **Aprendizaje supervisado:** Sus algoritmos se basan en la relación entre las entradas y las salidas. Por tanto, requiere de un proceso de clasificación previa muy posiblemente supervisado por un humano. Su objetivo final es el de obtener una salida para cada nueva entrada basándose en las relaciones obtenidas del conjunto de datos inicial.
2. **Aprendizaje no supervisado:** Se basa en algoritmos que aprenden de datos no etiquetados, es decir, solo existen entradas en los datos introducidos. Su objetivo es buscar patrones entre esos datos para agruparlos o asociarlos entre sí de alguna forma.
3. **Aprendizaje por refuerzo:** También llamado Reinforcement Learning (RL) , no pertenece a ninguno de los dos tipos anteriores ya que se diferencia del aprendizaje supervisado en que no están etiquetadas los datos y del no supervisado en que no intenta separar estos datos en grupos atendiendo a patrones. Se basa en un esquema en el que una máquina tiene que tomar decisiones o realizar acciones, las cuales se premian cuando acierta, mientras se penalizan los fallos que pueda cometer. [5]

Figura 2.2: Tipos de Aprendizaje Automático [4]

2.1.3. Aprendizaje Profundo

El Aprendizaje Profundo, más conocido como Deep Learning (DL) , es una rama del ML que pretende imitar cómo funciona el cerebro humano, reflejando la conexión que hay entre las neuronas para así poder replicar cómo las personas perciben y procesan la información. [6]

El DL es una forma de nombrar al aprendizaje llevado a cabo por redes neuronales profundas, las cuales son redes neuronales artificiales con varias capas ocultas entre las capas de entrada y salida. En el siguiente apartado se va a explicar qué son estas capas y cómo funcionan las redes neuronales.

2.1.4. Redes Neuronales

Las redes neuronales o Neural Networks (NN) son programas que se encargan de combinar unos parámetros dados para conseguir un cierto resultado. La complicación de esto está en cómo combinar esos parámetros. Cuando se entrena una red neuronal, realmente se está buscando la mejor combinación que se ajuste a los datos introducidos. Cuando ya está entrenada, esta es capaz de predecir o clasificar una nueva situación, en donde realmente está aplicando la mejor combinación conseguida en el entrenamiento previo.

Antes de entender el funcionamiento de una neurona artificial, se va a explicar a continuación cómo funciona una biológica. Su función principal es recibir, procesar y transmitir la información utilizando impulsos eléctricos. Se comunican entre ellas a través de la sinapsis. En este proceso, una neurona emisora genera neurotransmisores a partir de una descarga química que alcanzan a una neurona receptora, la cual se excitará o inhibirá dependiendo de la situación. En la figura 2.3, se representa una red neuronal biológica especificando las partes de las neuronas.

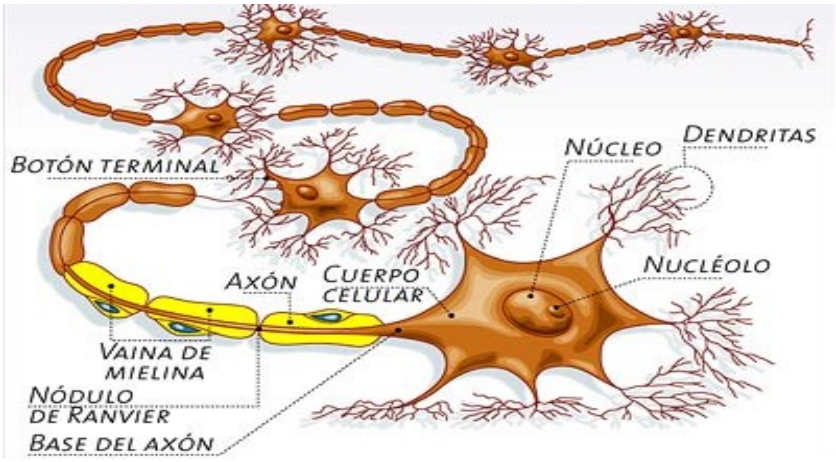


Figura 2.3: Partes de neurona biológica [7]

Las neuronas artificiales, las cuales intentan imitar el comportamiento de las neuronas biológicas, forman las NN artificiales [8]. Como se puede observar en la figura 2.4, estas neuronas están formadas por:

1. **Ramificaciones de entrada:** Son las entradas al nodo de la neurona y es por donde llega la información que procede de otras neuronas.
2. **Nodo:** Es la parte de la neurona donde la información es procesada.
3. **Ramificaciones de salida:** Es el lugar por el que se envía a otras neuronas la información procesada por el nodo.

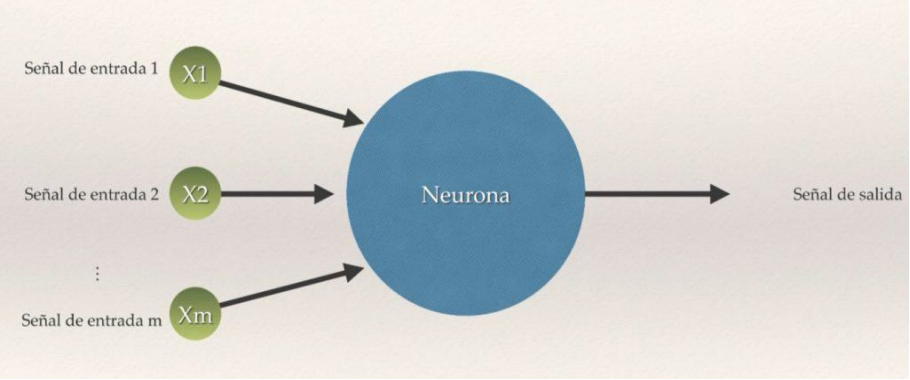


Figura 2.4: Partes de neurona artificial [8]

El primer modelo matemático de una neurona artificial como la que se acaba de explicar fue diseñado por Warren McCulloch y Walter Pitts en 1943. Este modelo, mostrado en la figura 2.6, se compone de:

1. **X o vector de entrada (inputs):** Contiene un valor para cada ramificación de entrada: $x_j \in X$.
2. **W o vector de pesos (weights):** Da un peso a cada ramificación de entrada, el cual determina la importancia de la entrada a la que está asociado: $w_j \in W$.
3. ***b* o sesgo (bias):** Parámetro que da preferencia a la activación de algunas neuronas frente a otras.
4. **$\sum$ o sumatorio (sum):** Es una de las funciones llevada a cabo por el nodo de la neurona y consiste en aplicar una regla de propagación. La más común es la siguiente: $\sum_{j=1}^{n} x_j \cdot w_j + b$
5. **Función de activación (activation function):** Es la otra función llevada a cabo por el nodo de la neurona. Se encarga de activar o desactivar la salida de la neurona. Hay distintos tipos de función de activación como los que se muestran en la figura 2.5.

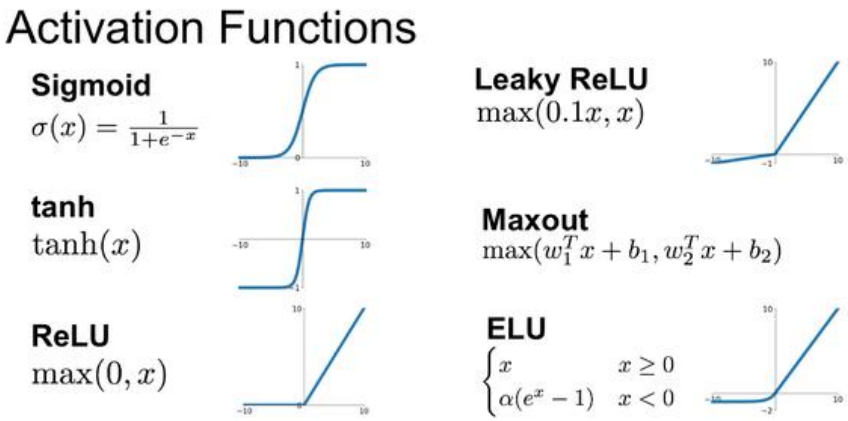


Figura 2.5: Tipos de funciones de activación [9]

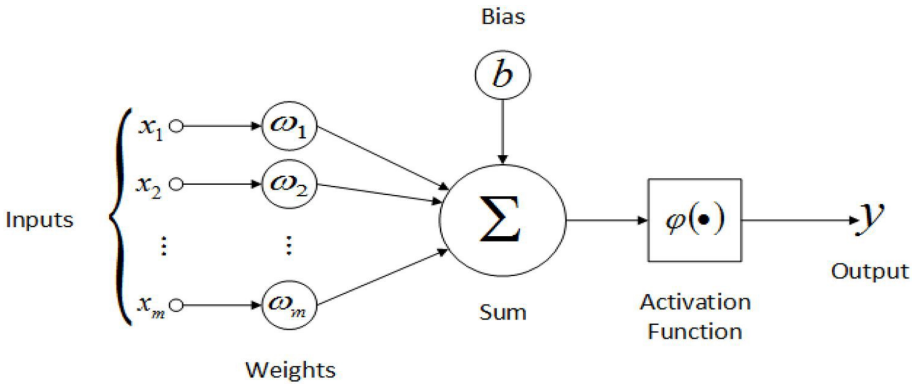


Figura 2.6: Parte matemática de una neurona artificial [10]

Por tanto, una red neuronal está formada por neuronas como la descrita anteriormente, sin embargo no todas tienen ramificaciones de entrada y ramificaciones de salida como se puede observar en la figura 2.7. Las neuronas que forman la capa de entrada o *input layer* (las rojas en la figura) no tienen ramificaciones de entrada ya que la información que reciben estas proviene directamente desde el exterior. En cambio, las neuronas que forman la capa de salida o *output layer* (las azules en la figura) no tienen ramificaciones de salida ya que contienen la información final. Por último, las neuronas amarillas, que son las que forman las capas ocultas o *hidden layers*, tienen ambas ramificaciones.

Entrenar una red neuronal consiste en ajustar cada uno de los pesos en cada entrada de las neuronas para que la salida se ajuste lo máximo posible a los datos que conocemos. Una NN puede ser simple o profunda dependiendo del número de las mencionadas capas ocultas, como también se puede observar en la figura 2.7.

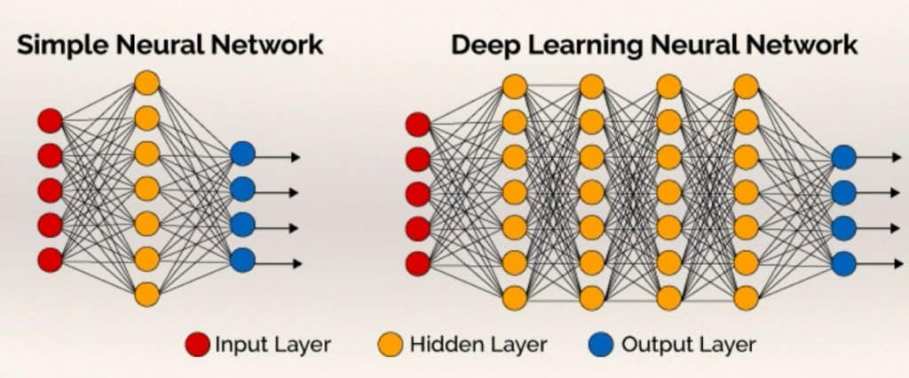


Figura 2.7: Redes neuronales [8]

2.1.5. Tipos de Redes Neuronales

En esta sección se van a detallar los tipos de redes neuronales más conocidos.

Red neuronal Monocapa o Perceptrón simple

Esta NN solo tiene una capa de neuronas de entrada y otra de neuronas de salida, por tanto, no existen las *hidden layers* en este modelo. Se puede ver un ejemplo de ello en la figura 2.8.

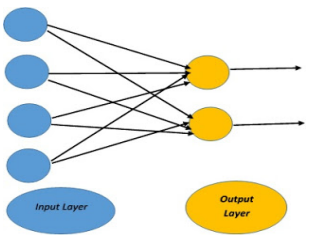


Figura 2.8: Perceptrón simple [11]

Red neuronal Multicapa o Perceptrón multicapa

El Perceptrón multicapa se diferencia del monocapa en que tiene capas intermedias o *hidden layers*. Teniendo en cuenta el número de conexiones, esta red puede estar totalmente conectada, es decir, que todas las neuronas de una capa estén conectadas a las de la siguiente capa y así sucesivamente, o puede estar parcialmente conectada si la anterior condición no se da. En la figura 2.9 se muestra un ejemplo de red neuronal multicapa totalmente conectada.

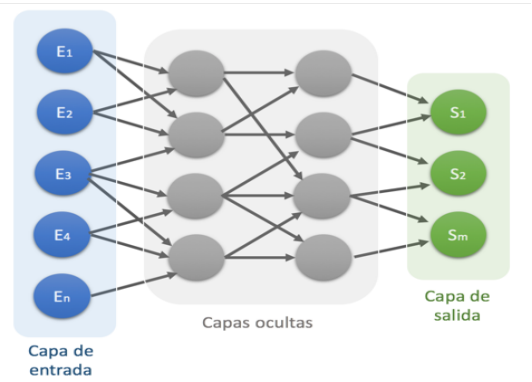


Figura 2.9: Perceptrón multicapa [12]

Red neuronal Convolucional

Esta red, también llamada Convolutional Neural Network (CNN) , es una variación del perceptrón multicapa ya que está estructurada de la misma manera pero utiliza matrices bidimensionales, las cuales son muy útiles para tareas de visión artificial, como la clasificación de imágenes.

Para poder entender cómo funciona esta NN , primero hay que entender que una imagen sin color es una matriz de píxeles y que el valor de estos está entre 0 y 255, en cambio una imagen a color está formada por 3 matrices de píxeles, una de cada color Red, Green and Blue (RGB) como se aprecia en la figura 2.10. Esta matriz o matrices serían la entrada de la CNN , pero existe un pre-procesamiento en el que los píxeles son normalizados para que sean valores entre 0 y 1, por tanto se dividen entre 255.

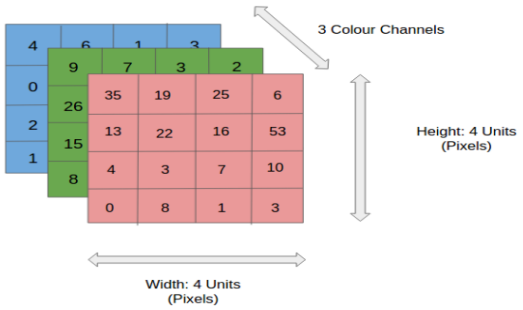


Figura 2.10: Matrices de píxeles en imagen con color [13]

Después de conseguir la entrada normalizada, vienen las convoluciones, las cuales consisten en agrupar píxeles cercanos de la entrada y hacer el producto escalar de ellos contra una pequeña matriz llamada **Kernel** o filtro, lo cual reducirá notablemente las dimensiones. En caso de ser a color, habría un kernel para cada capa RGB , pero después se sumarían los resultados de las 3 capas resultantes y quedaría como si fuese 1 sola. Hay varios kernels o filtros, y estos se va desplazando por la matriz de entrada de tantas en tantas posiciones como indique el *stride*. Un ejemplo gráfico de cómo funcionan estas convoluciones es mostrado en la figura 2.11.

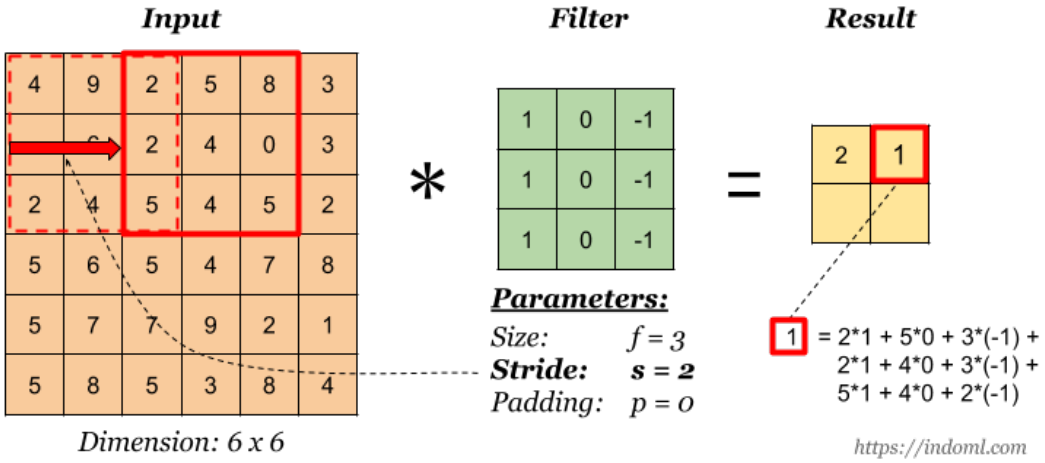


Figura 2.11: Cómo funcionan las convoluciones [14]

La siguiente fase es el muestreo, el cual consiste en reducir el tamaño de las capas de neuronas preservando las características más importantes. Sin este paso, la capacidad computacional podría ser demasiado grande. Hay diversas formas de realizar el muestreo, sin embargo, la más importante y más usada con diferencia es el Max-pooling. Consiste en seleccionar el píxel más alto de una dimensión escogida, un ejemplo es mostrado en la figura 2.12.

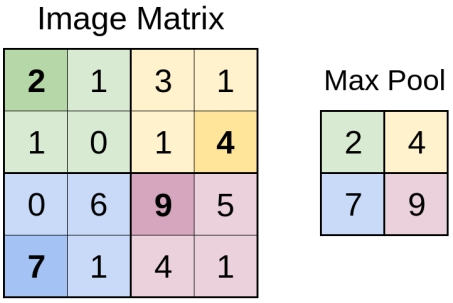


Figura 2.12: Ejemplo de Max-Pooling [15]

Hasta aquí la CNN solo sería capaz de reconocer características primitivas como líneas o curvas, sin embargo, cuantas más repeticiones de estas fases, sería capaz de reconocer más patrones en una imagen. En la figura 2.13 se representa un ejemplo con las fases de convoluciones que puede haber.

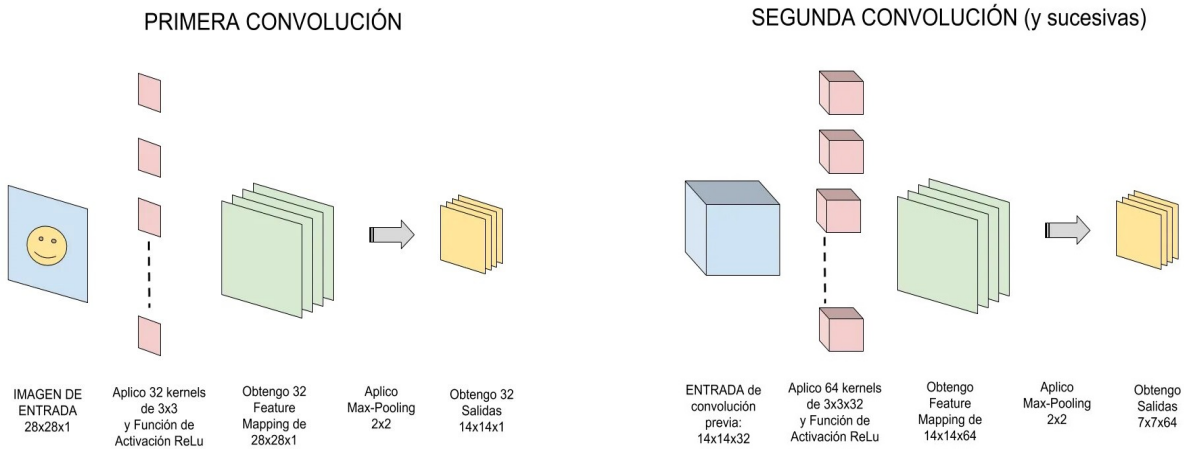


Figura 2.13: Ejemplo de convoluciones [13]

Después de todas estas fases, se ponen los filtros que hayan salido en una sola dimensión para que se forme una sola capa de neuronas *tradicionales*, que es llamada *fully connected*. A esta capa se la aplica una función llamada *softmax* que se encarga de conectarla con la capa de salida, la cual tendrá la misma cantidad de neuronas que las clases en las que se esté clasificando. Esta capa de salida tiene el formato *one-hot-encoding*, lo que significa que para cada clase la salida será 0 (si predice que la entrada no corresponde con esa clase) o 1 (si predice que la entrada corresponde con esa clase). Además, la función de *softmax* también indica la probabilidad de que sea una clase u otra. Un buen ejemplo de la arquitectura de las CNN es mostrado en la figura 2.14.

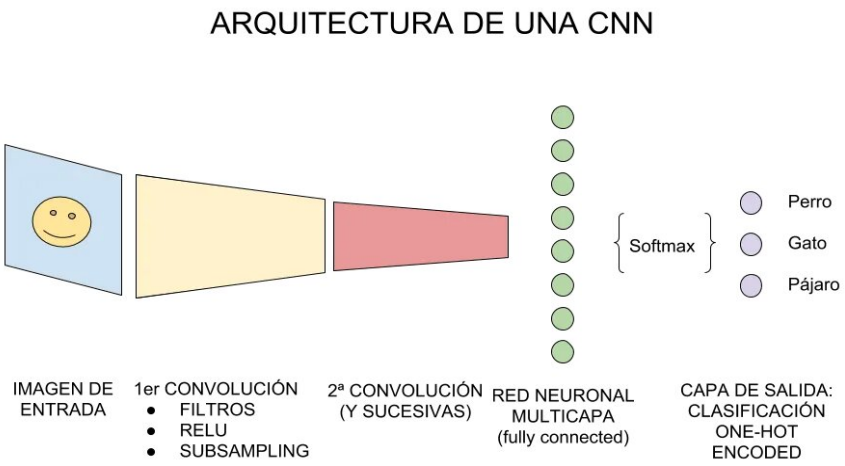


Figura 2.14: Arquitectura de una CNN [13]

Red neuronal recurrente

La red neuronal recurrente o Recurrent Neural Network (RNN) no tiene una estructura de capas como las otras redes neuronales, sino que existen conexiones arbitrarias permitiendo crear ciclos e incluso dotar a la red de memoria. Esto las hace especialmente atractivas para el procesamiento de secuencias, tales como el análisis de textos, sonido o vídeo.

La memoria que se ha mencionado que tiene una RNN consiste en que después de producir una salida, esta se tiene en cuenta a la hora de tomar una decisión para la siguiente salida. Para explicar esto más visualmente, en la figura 2.15 se muestra un modelo en el cual, al principio, la entrada es X_{t-1}. Esto genera una salida (h_{t-1}) que, a su vez, será la entrada junto a X_t del siguiente paso, y así sucesivamente [16]. En esta figura se puede observar un ejemplo muy simple en el cual se observa que la RNN está formada por varios módulos de redes neuronales y que cada módulo tiene una única capa oculta con la función de activación tangente hiperbólica (vista anteriormente).

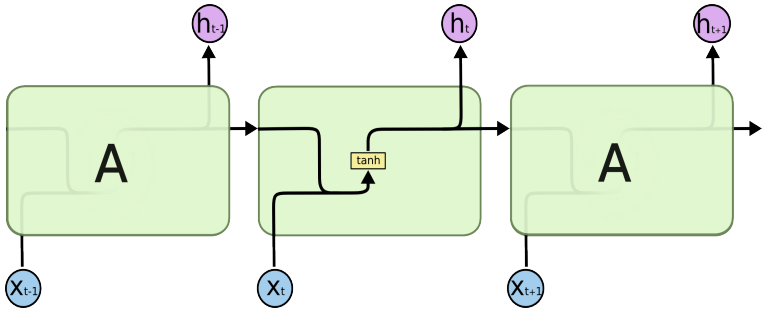


Figura 2.15: Funcionamiento de una RNN [17]

Estas redes se pueden implementar de varias formas, pero la versión más conocida y usada es el modelo Long Short Term Memory (LSTM) . Este tipo de RNN hace que la memoria recuerde más fácilmente los datos por un largo período de tiempo. Para explicar el funcionamiento de LSTM se va a mostrar en la figura 2.16 cómo está formado cada módulo. Se diferencia en el ejemplo sencillo de una RNN de la figura 2.15 en que existen 4 capas y no 1, las cuales además actúan de una manera especial.

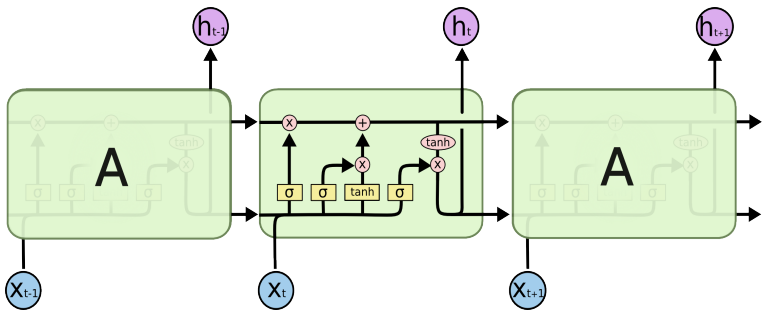
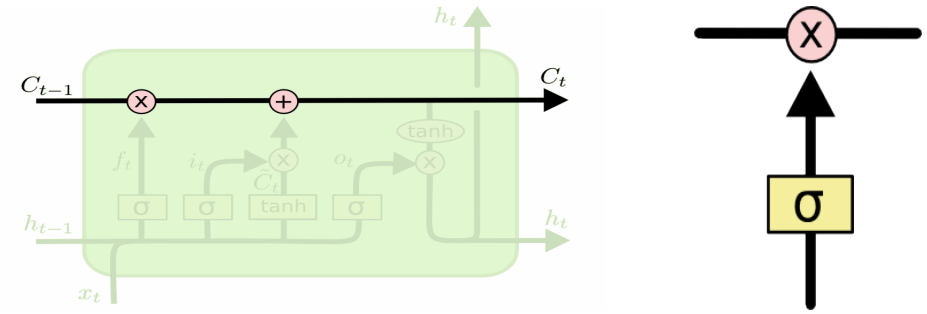


Figura 2.16: Esquema LSTM [17]

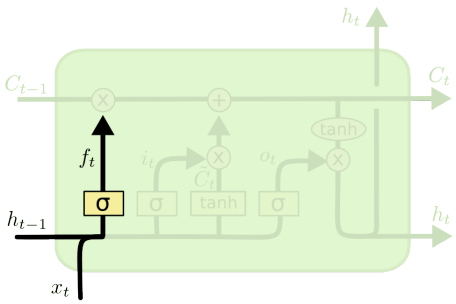
Para empezar a describir este modelo, primero hay que fijarse en la parte clave que es la *cell state* (mostrada en la figura 2.17a), la cual es una especie de cinta transportadora de información que permite el flujo de datos pero con cambios mínimos. Se puede añadir o eliminar esta información con el equivalente a las puertas lógicas (figura 2.17b), las cuales están formadas por una función de activación y la operación de multiplicación. La función de activación es la sigmoidea que tiene como salida un intervalo de 0 a 1, donde 0 no deja pasar información y 1 deja pasar toda la información. [18]



(a) Cell state (b) Puerta lógica

Figura 2.17: LSTM-Cell state [17]

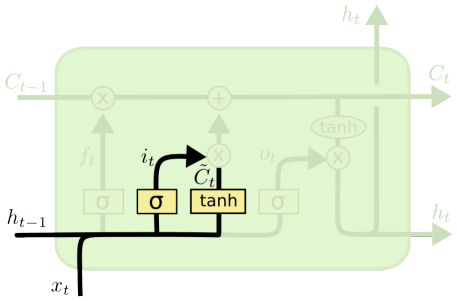
El primer paso que se lleva a cabo en la LSTM consiste en decidir qué información se desecha de la *cell state*. Esto es llevado a cabo en la primera capa donde las entradas son X_t y $h_(t-1)$, a las cuales se las multiplica por el peso W_f correspondiente, teniendo en cuenta el bias (b_f) que se suma y a todo esto se le aplica la función de activación sigmoidea. Se observa esto que se acaba de explicar en la figura 2.18.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figura 2.18: LSTM - Paso 1 [17]

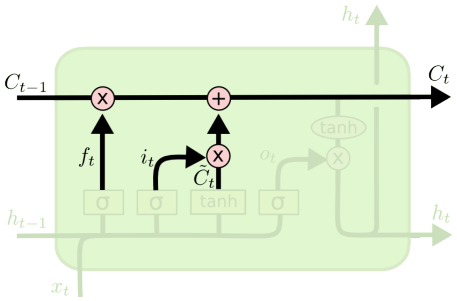
El segundo paso es decidir qué información se almacena en la *cell state*. Para ello, lo primero es decidir qué valores se van a actualizar (i_t), lo cual es llevado a cabo por una función como la del paso anterior, pero con sus pesos y bias correspondientes. Deespués, se crea un vector con los nuevos candidatos ($\widetilde{C}_t$), utilizando otra función de activación, en este caso, la tangente hiperbólica. Este paso es reflejado en la figura 2.19.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figura 2.19: LSTM - Paso 2 [17]

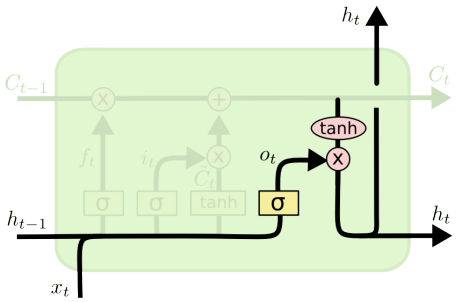
El tercer paso es aplicar las decisiones y valores conseguidos en los 2 anteriores pasos, actualizando así la *cell state* pasando de ser C_{t-1} a C_t. Para ello, se multiplica el antiguo estado C_{t-1} por f_t, olvidándose la información que se ha decidido en el primer paso. A ello se suman los nuevos valores candidatos ($\widetilde{C}_t$) multiplicados por la decisión de cuál de estos valores realmente se van a actualizar (i_t). Esta operación se aprecia más visualmente en la figura 2.20.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figura 2.20: LSTM - Paso 3 [17]

Por último, se necesita decidir qué es lo que va realmente a la salida. Para ello primero se ejecuta una capa con una función de activación sigmoide (o_t) que decidirá qué información se quiere conservar. Los datos conseguidos en la *cell state* del paso anterior (C_t) se multiplican por esto y así se consigue filtrar lo relevante hacia la salida como se puede observar en la figura 2.21.



$$o_t = \sigma\left(W_o \; [\, h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figura 2.21: LSTM - Paso 4 [17]

2.1.6. Usos de Redes Neuronales en el ámbito de la música

En cuanto a la clasificación de cualquier tipo de sonido, las NN más adecuadas serían las CNN. Esto se debe a que el sonido se puede visualizar de varias formas, las cuales se pueden convertir en imágenes para ser tratadas por las CNN. La forma más tipica es la representación de la onda de sonido mostrada en la 2.22, donde se representa el tiempo en el eje x y la amplitud en el eje y.

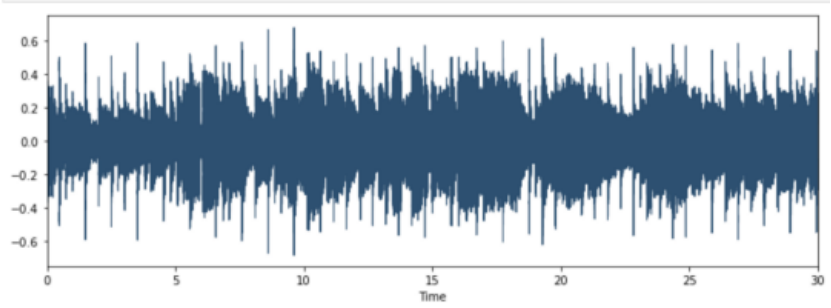


Figura 2.22: Ondas del sonido [19]

Otra de las formas de representar el sonido y con la cual, según algunos estudios como [20], las CNN funcionan mejor ya que además de representar tiempo y amplitud, se representa la frecuencia, son los espectrogramas (2.23). En ellos se representa la frecuencia (eje y) frente al tiempo (eje x) y los colores indican la amplitud.

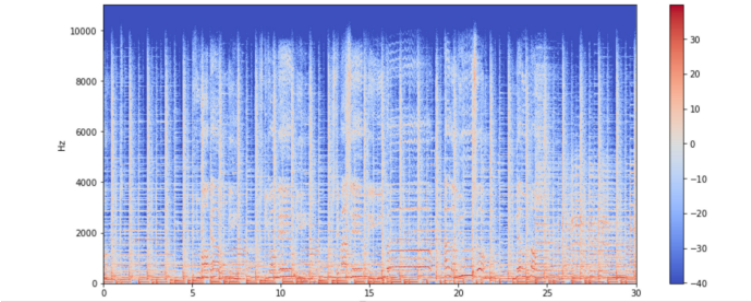


Figura 2.23: Espectrograma [19]

Por tanto, si se tuviese que elegir una red neuronal para clasificar música, como es el caso, lo más adecuado sería utilizar una CNN con espectrogramas convertidos a imágenes como datos de entrada.

Sin embargo, no es lo mismo **predecir** música que **generarla**. Para esta segunda tarea, las NN más utilizadas son las RNN ya que tienen en cuenta las entradas anteriores. Es decir, para generar una nueva nota musical, tienen en cuenta las anteriores. Esto es necesario, ya que en una misma pieza musical hay que mantener una determinada melodía, con una determinada armonía y con un determinado ritmo (conceptos explicados en la siguiente sección), cosa que solo se puede conseguir teniendo presente la parte anterior de la composición.

2.2. Música

Para la automatización de la generación de la música hay que tener claros algunos conceptos musicales que se describen en esta sección. La definición de música que mejor se conecta a la temática de esta investigación es: Arte que consiste en organizar sensible y lógicamente una combinación coherente de sonidos y silencios utilizando los principios fundamentales de la melodía, la armonía y el ritmo. [21]

2.2.1. Elementos de la música

Los elementos de la música son la melodía, la armonía y el ritmo [22]:

1. **Melodía:** Es el conjunto de sonidos y silencios que suenan consecutivamente y poseen un significado propio.
2. **Armonía:** Es el resultado de combinar dos o más notas musicales, lo cual puede ser más o menos agradable (armónico) para el oído. Un conjunto de sonidos concordantes reproducidos al mismo tiempo se llama acorde.
3. **Ritmo:** Es la unión entre la música y el tiempo. Además, se considera el elemento organizativo de la música.

2.2.2. Características de la música

La música es transmitida a partir de sonidos, es decir, se propaga por el aire hasta llegar al oído. Estos sonidos tienen 4 parámetros principales:

Altura

Es la cualidad sonora que permite distinguir sonidos agudos y graves. Esto está relacionado con la frecuencia de la onda de los sonidos.

Es importante conocer cómo esos sonidos graves o agudos son reflejados en el lenguaje musical. Las notas musicales son la equivalencia a las diferentes frecuencias de los sonidos. Un ejemplo de estas equivalencias se puede observar en la figura 2.24.

Otro concepto relacionado con la altura es la tonalidad que se define como el tono predominante a partir del cual una obra musical se estructura. Las tonalidades se diferencian entre sí porque cada una genera una sensación distinta. Además, cada tonalidad presenta unas alteraciones propias, indicadas al comienzo del pentagrama en un lugar llamado armadura.

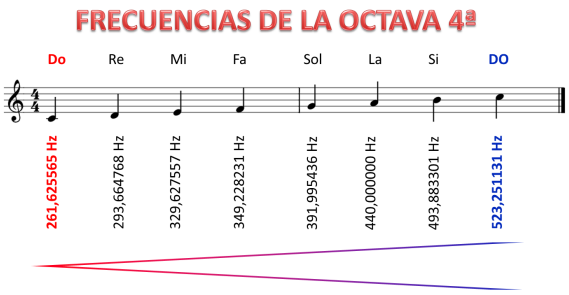


Figura 2.24: Relaciones entre frecuencias y notas musicales [23]

Duración

Característica estrechamente relacionada con el ritmo. Es el tiempo que permanece un sonido siendo emitido.

Dentro del lenguaje musical, los elementos que denotan la duración de las notas musicales son las figuras musicales. En la figura 2.25 se muestran estas figuras musicales y el valor de duración que tiene cada una.

Nombre de la figura	Figura	Nombre del silencio	Silencio	Valor
Cuadrada		Silencio de cuadrada		
Redonda		Silencio de redonda		1
Blanca		Silencio de blanca		2
Negra		Silencio de negra		4
Corchea		Silencio de corchea		8
Semicorchea		Silencio de semicorchea		16
Fusa		Silencio de fusa		32

Figura 2.25: Figuras musicales [23]

Intensidad

Esta cualidad permite diferenciar los sonidos fuertes de los débiles, los cuales están relacionados directamente con la amplitud de las ondas producidas por los sonidos.

Los elementos que reflejan la intensidad en el lenguaje musical se llaman matices. En la figura 2.26 se muestran algunos de ellos.

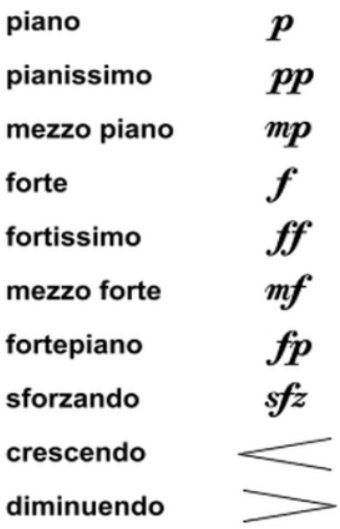


Figura 2.26: Matices musicales [24]

Timbre

Se centra en la forma de las ondas de los sonidos y es la cualidad que permite diferenciar unos instrumentos de otros.

2.2.3. MIDI

Musical Instrument Digital Interface (MIDI) es un protocolo que permite a instrumentos musicales electrónicos y dispositivos comunicarse entre sí para intercambiar información de piezas musicales.

La música, como cualquier tipo de audio almacenado en un ordenador o móvil, se suele guardar en formato MP3 o similares, sin embargo, es posible almacenar la información de una pieza musical, es decir, almacenar una especie de partitura. Esta información se almacena en un archivo de extensión .mid, el cual es interpretado por el dispositivo y puede ser reproducido. Explicándolo de otra forma, es como si fuese una partitura musical que puede ser interpretada por un dispositivo o un instrumento musical electrónico.

En el ámbito de la generación de música artificial, este formato puede llegar a resultar clave, ya que a la hora de introducir canciones a una RNN, esta red debe poder identificar cada nota musical y algunos de los parámetros que solo constan en estas partituras digitales. Los parámetros que son más relevantes a la hora de introducir un MIDI a una RNN son la altura y la duración. Esto se debe a que, actualmente, las RNN que componen las herramientas más avanzadas en la generación de música, son capaces de ir generando notas musicales (**altura**) con una **duración** determinada para que tenga sentido. Sin embargo, esa pieza generada es con un solo instrumento (no hay **timbre**) y esas notas suenan todas con la misma **intensidad**, aparte, ese único instrumento (Acoustic Gran Piano) solo es capaz de generar melodías monofónicas.

2.3. Web Scraping

El WS consiste en la recolección de datos y contenido de un sitio web. Esta técnica se suele utilizar en el contexto de la IA para entrenar y validar los modelos con los datos recogidos. Como se puede observar en la figura 2.27 consta de 3 pasos:

1. Se estudia el código HTML de la página web objetivo.
2. De ese código HTML se extraen los datos que sean de interés.
3. Esos datos extraídos se pueden almacenar estructuradamente en bases de datos.

Figura 2.27: Web scraping [25]

2.4. Estudios similares

En esta sección se compara el presente TFG con investigaciones similares llevadas a cabo por otras personas. En la tabla 2.1 se muestra un resumen de esta comparación indicando los temas que aborda y las tecnologías utilizadas cada una de ellas.

Para empezar, los temas principales de este trabajo son la composición de BSO utilizando RNN , pudiendo elegir el género cinematográfico del que se desea obtener la pieza musical, y la investigación sobre la posibilidad de clasificar las BSO en estos géneros de películas (tanto con CNN como con la participación de personas). Sobre la investigación de asociar géneros cinematográficos a música, no se ha encontrado mucho y las investigaciones comparadas son las que más se pueden asimilar. Por tanto, las comparaciones son con proyectos que utilizan las mismas tecnologías y que abordan temas similares pero no iguales.

La Investigación 1 (I1) [18] cuenta con mucho detalle las tecnologías que ha utilizado, en este caso, una herramienta que utiliza RNN , específicamente LSTM . El objetivo principal de este estudio es investigar sobre la creación de música utilizando muchas canciones de un artista como entrenamiento y, posteriormente, con la participación de personas, valorar si las composiciones llegan a ser tan buenas como para pensar que pueden ser de ese artista. Hace 3 experimentos con 3 artistas distintos.

La Investigación 2 (I2) [20] investiga sobre la posibilidad de que una CNN sea capaz de distinguir el género musical de una canción utilizando espectrogramas.

La Investigación 3 (I3) [26] estudia la clasificación de las películas según su género cinematográfico utilizando CNN tanto para el audio como el vídeo. Utiliza como entrenamiento trailers en vez de las películas enteras para que no requiera un poder computacional desorbitado.

Temas abordados o tecnologías utilizadas	**Este TFG**	**I1** [18]	**I2** [20]	**I3** [26]
Parte cinematográfica	Sí	No	No	Sí
Parte musical	Sí	Sí	Sí	No
Tratamiento de sonido	Sí	Sí	Sí	Sí
Tratamiento de vídeo	No	No	No	Sí
Utilización de CNN	Sí	No	Sí	Sí
Utilización de RNN y LSTM	Sí	Sí	No	No
Utilización de espectrogramas	Sí	No	Sí	No

Tabla 2.1: Tabla resumen de la comparación de temas abordados y tecnologías utilizadas en investigaciones similares.

Este trabajo va a operar con canciones para diferentes géneros cinematográficos en lugar de con canciones de diferentes estilos (rock, clásica...) como I2 , y tampoco con canciones de artistas concretos como I1 . Aunque I3 trabaja con géneros cinematográficos, no trata la música, sino que trata el audio (voz, ruidos de fondo, música...) y el vídeo por lo que tampoco es idéntica a la presente investigación. En cuanto a la IA utilizada hay más similitud, este trabajo utiliza CNN , que también son usadas en la I2 y en la I3 . Además, las RNN , y más en concreto las LSTM , también son utilizadas para la generación de música en este TFG , las cuales son utilizadas de forma similar en I1 .

3

Desarrollo

En este capítulo se explican los pasos llevados a cabo en esta investigación con todo tipo de detalle, describiendo las herramientas y cualquier elemento que se haya utilizado para llegar a los resultados obtenidos. Antes de explicar cómo se llevó a cabo, hay que comentar que todo el código de este TFG está escrito en Python, el cual es un lenguaje de programación de código abierto ampliamente usado para tareas de Machine Learning. Todo este código está en el GitHub [27]: `https://github.com/albertonavas99/TFG-Inteligencia-artificial-en-bandas-sonoras/`.

3.1. Extracción de datos

El primer paso realizado en esta investigación fue estudiar los diferentes datos que se podían obtener para los objetivos propuestos.

En Internet, existen infinidad de datos que se pueden recopilar con diversas técnicas. Entre ellas, destaca el WS que se ha explicado anteriormente en el capítulo 2. Sin embargo, existe otra técnica, la cual solo ofrecen algunos sitios web y es el uso de una Application Programming Interfaces (API) pública. Una API pública explica detalladamente cómo hacer una petición a un sitio web con unos determinados parámetros para que te devuelva los resultados deseados.

Para la realización de este trabajo se han llevado a cabo varios estudios de los datos que se pueden obtener a través de Internet, de los cuales se ha obtenido una base de datos final que cuenta con 565 MIDIs y apróximadamente 60.000 MP3.

3.1.1. Primer estudio de datos

Realizando esta investigación, en un primer estudio se encontró la web de *themoviedb* [28], donde cada película contaba con campos muy útiles para este estudio (nombre de película y géneros cinematográficos) y, además, contaba con una API pública con la cual se podrían obtener estos datos de una forma sencilla. Por tanto, se decidió investigar más a fondo estos datos ofrecidos, entre los que se encontró un enlace al tráiler de cada película. A primera impresión, muchos de esos trailers tenían

una canción de fondo, la cual se podía intentar estudiar para diferenciar el género cinematográfico. Por lo cual, se llevó a cabo la extracción de los datos de cada película y la descarga de los audios de los trailers. Sin embargo, tras muchas pruebas de clasificación por género y un exhaustivo estudio del audio de los trailers, incluso separando la parte musical de la voz, se llegó a la conclusión de que la base de datos obtenida no era coherente y no daría lugar a una conclusión acertada. Esto se debía a que muchos de los trailers descargados no tenían mucha música, e incluso algunos eran de personas manteniendo una conversación sobre la película sin ninguna música de fondo. Al ser tantos datos (alrededor de 8500), hubiese sido un trabajo demasiado laborioso para una persona eliminar todas las muestras que pudiesen corromper la investigación, por lo cual se decidió cambiar de estrategia. El código de este primer estudio se encuentra en el archivo **extraccion_datos_estudio1.py** del GitHub [27].

3.1.2. Segundo estudio de datos

En un segundo estudio de cómo obtener los datos requeridos para el trabajo, se buscaron datos que fuesen solo BSO sin efectos de sonido ni voces, pero además, se tuvo en cuenta que hubiese una versión MIDI de esas pistas para que valiesen también para la parte de generación de melodías y no solo para la parte de clasificación. En el primer intento (comentado en la subsección 3.1.1), se estudió la viabilidad de conseguir transformar los audios de los trailers de .mp3 a .midi, y, aunque se puede hacer esta conversión con diversas herramientas, el resultado no es para nada satisfactorio.

En este segundo estudio, se encontró una base de datos de MIDIs [29], la cual además de tener una sección de bandas sonoras, las pistas están en formato .mp3 y .midi. Sin embargo, esta página web no tiene una API pública por lo que se tuvo que realizar la técnica del WS .

Para llevar a cabo el WS se ha utilizado la librería requests [30], la cual permite realizar peticiones y obtener el HTML de la respuesta en texto plano. Además, se ha utilizado también la librería Beautifulsoup [31], la cual hace más fácil acceder a los elementos deseados de esta respuesta en código HTML .

En este caso, lo primero que se quería conseguir era un listado con todas las pistas de BSO de películas de la web y las URLs de descarga, tanto del MIDI como del MP3. Para ello, se recorrieron todas las páginas de la web en las que había BSO , o como lo llama la página *movie themes*. En la figura 3.1, se pueden observar las distintas páginas a las que se puede acceder por letra y los títulos de las películas de los que se puede descargar su BSO cuando se hace click en ellos. Además, si uno se fija en la URL de la figura, para conseguir acceder a cada letra, simplemente hay que variar el final de esta, poniendo la letra deseada, en este caso, se muestran todas las películas que comienzan por la *a*.

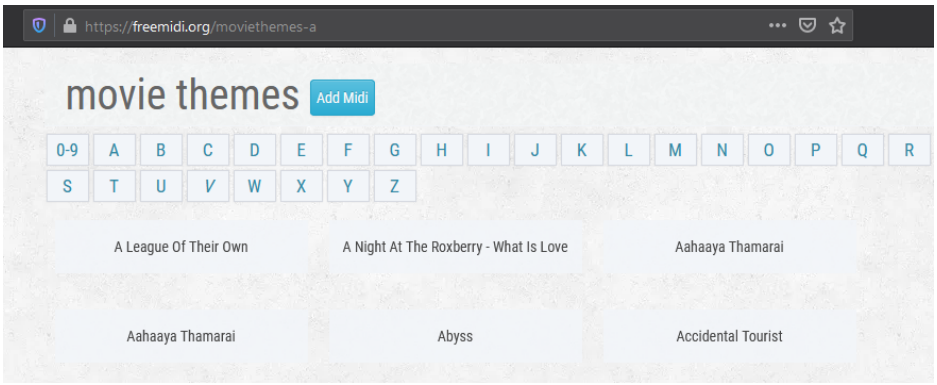


Figura 3.1: Página web freemidi.org [29]

Una vez recogido un diccionario con los títulos de las películas como clave y las URLs de descarga como valor, se necesitaban conseguir los géneros de esas películas, ya que esta página no los va a proporcionar. Para ello, se hizo uso de otra página web que cuenta con una base de datos de películas y sus géneros [32].

Para poder acceder a la página de una película específica donde conseguir sus géneros cinematográficos, primero hay que encontrar esa página. Esto se puede conseguir realizando una petición a la URL de búsqueda de películas de esta web, la cual se forma con el título de la película correspondiente separada por símbolos de +. La primera entrada de la búsqueda contiene un link a la página de la película, el cual se almacena. Esto se puede observar en la figura 3.2.

Figura 3.2: Página de búsqueda de películas imdb.com [32]

Ya con el link de una página de una película específica, se puede realizar una petición y, con la respuesta HTML , se extraen con WS los géneros cinematográficos a los que pertenece esta.

El siguiente paso fue descargar los .midi y .mp3 de los cuales se tenía ya la URL almacenada. Para esto, las peticiones a esas URLs hubo que hacerlas con unas cabeceras específicas para que el servidor pudiese detectar una sesión en la cual poder almacenar cookies. Una vez obtenida la respuesta, hubo que almacenarla escribiendo un fichero en binario y dándolo la extensión que correspondiese (.mid o .mp3).

Por lo tanto, se consiguió obtener una base de datos de 565 archivos MIDI y 565 archivos MP3, etiquetados cada uno con sus géneros cinematográficos correspondientes. Se detalla, en la figura A.1 del Apéndice A, un flujograma de este segundo estudio. El código de este segundo estudio se encuentra en el archivo **extraccion_datos_estudio2.py** del GitHub [27].

3.1.3. Tercer estudio de datos

Se siguió investigando en un tercer estudio los datos que hay accesibles en Internet y se llegó a la conclusión de que los MIDIs eran archivos muy difíciles de encontrar y difícilmente se podría aumentar esa base de datos. Sin embargo, los archivos de audio normal (MP3) para la parte de clasificación son mucho más fáciles de encontrar, así que se decidió aumentar la base de datos de este tipo de archivos.

Para ello, se encontró una página llamada soundtrackcollector [33], donde de forma similar al segundo estudio, se tuvo que sacar un listado con todos los títulos de películas que tienen BSO , recorriendo las diferentes páginas de la web. También, al igual que en el segundo estudio, se buscarán los géneros en IMDB [32]. Sin embargo, lo que ocurre en esta ocasión, es que la página de soundtrackcollector [33] no proporciona ningún enlace de descarga ni nada parecido. Por tanto, se almacenaron, de cada película con banda sonora, su título y el compositor de la misma. Esto se debe a que se realizó un tercer cruzado de datos, es decir, se utilizó una API de youtube music [34] para buscar las obras musicales a partir del título de la película y el compositor de la banda sonora. Una vez conseguido el link a la canción de youtube, ya se pudo descargar con youtube-dl [35] (librería que permite descargar audio y vídeo de youtube) la pieza musical en formato MP3.

Tras finalizar esta extracción de datos, se consiguieron alrededor de 60.000 bandas sonoras en formato MP3. El código de este estudio se encuentra en el archivo **extraccion_datos_estudio3.py** del GitHub [27].

3.1.4. Etiquetado final de los datos

Una vez conseguida la base de datos final que cuenta con 565 MIDIs y apróximadamente 60.000 MP3 de BSO , además de estar cada archivo relacionado con los géneros cinematográficos correspondientes, se puede decir que están todos los datos etiquetados. Sin embargo, hay multitud de géneros cinematográficos que se parecen entre sí y algunos que se pueden quitar, esto es para que al realizarse los experimentos en las NN , haya menos carga computacional y pueda dar lugar a menos confusiones.

Para ello, se van a dejar únicamente las películas con los siguientes géneros cinematográficos: Comedia, Drama, Deporte, Música, Romance, Misterio, Ciencia Ficción, Suspense, Fantasía, Oeste, Acción y Terror. Este proceso está implementado en el archivo **reduccion_generos.py** del GitHub [27].

3.2. Generación de composiciones

Una vez clasificadas las BSO por sus géneros cinematográficos, en esta sección se intenta llevar a cabo la generación de nuevas composiciones para películas, basándose en el género de estas. En esta sección solo se utilizarán los MIDIs y no los MP3 que se utilizarán en la parte de clasificación 3.3.

3.2.1. Herramientas utilizadas

Para llevar esto a cabo, el primer paso es elegir la red neuronal que se va a utilizar. Como se indicó en 2.1.6, para la generación de música, hay que tener en cuenta las anteriores notas a la que se va a generar. Por esta razón, se va a utilizar un proyecto llamado Magenta [36] que hace uso de las RNN, más concretamente de las LST. Este proyecto, lanzado por Google, es de código abierto y está realizado con la librería de TensorFlow. Magenta se centra en el desarrollo de nuevos algoritmos de Deep Learning, los cuales son utilizados para la música y el dibujo. También resultan muy interesantes diferentes modelos pre-entrenados que proporciona este proyecto. El código de este proyecto está en un repositorio de GitHub [37].

Entre los modelos que proporciona Magenta, el que mejores resultados da, como se indica en el estudio [18], es el *Melody RNN Attention* y, por tanto, es el utilizado en este trabajo. Este modelo es especialmente atractivo porque es capaz de recordar notas más alejadas en el tiempo, y por tanto, da lugar a un contornos melódicos más largos. Con esta *memoria*, este modelo tiene en cuenta la altura y duración de las notas anteriores, para que la nota generada sea melódica con las notas anteriores.

3.2.2. Entrenamiento del modelo *Melody RNN Attention*

Para llegar a conseguir el objetivo de generar una melodía monofónica de un género cinematográfico concreto, primero hay que separar cada MIDI en la carpeta de su género correspondiente. Esto es sencillo, ya que después del etiquetado explicado en la sección anterior 3.1, simplemente hay que hacer un script (el cual está en **separacion_generos.py** del GitHub de este trabajo [27]) que mueva cada archivo a su carpeta correspondiente.

Una vez organizados todos los MIDIs, se descargó el modelo pre-entrenado *Melody RNN Attention*, del cual se hicieron 12 copias, una por género, para que cada modelo sea entrenado con los MIDIs de un solo género.

El siguiente paso fue convertir los MIDIs en *Note Sequences*, que son unos mecanismos que serializan o deserializan las estructuras de datos de los objetos para reducir el tamaño de los MIDIs y que se puedan almacenar o ser enviados, además de aumentar el rendimiento a la hora de trabajar con ellos. Esto se hace con el comando *convert_dir_to_note_sequences*, el cual ejecuta código del

proyecto Magenta y por tanto solo funcionará cuando se haya importado este repositorio de manera local. Este comando se ejecuta con los siguientes argumentos:

- ***–input_dir*:** Después del cual se establece el directorio donde se encuentran todos los MIDIs que se quieren convertir en *Note Sequences*.
- ***–output_file*:** Después del cual se establece la ruta y el nombre del archivo con extensión .tfrecord que se va a generar.
- ***–recursive*:** Establece que se haga la conversión de todos los archivos que estén en el directorio indicado en *–input_dir*.

Un ejemplo del funcionamiento de este comando es el mostrado en la figura 3.3, donde se muestra la salida de la conversión de 6 MIDIs en un archivo llamado *notesequences.tfrecord*.

```
$convert_dir_to_note_sequences --input_dir=tmp\\MIDIs --output_file=tmp\\notesequences.tfrecord --recursive > conv.txt
2021-06-05 18:03:00.570344: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library cudart64_110.dll
INFO:tensorflow:Converting files in 'tmp\\MIDIs\'.
I0605 18:03:06.179393  8676 convert_dir_to_note_sequences.py:83] Converting files in 'tmp\\MIDIs\'.
INFO:tensorflow:Converted MIDI file tmp\\MIDIs\10.mid.
I0605 18:03:06.592330  8676 convert_dir_to_note_sequences.py:153] Converted MIDI file tmp\\MIDIs\10.mid.
INFO:tensorflow:Converted MIDI file tmp\\MIDIs\12 Monkeys.mid.
I0605 18:03:06.775839  8676 convert_dir_to_note_sequences.py:153] Converted MIDI file tmp\\MIDIs\12 Monkeys.mid.
INFO:tensorflow:Converted MIDI file tmp\\MIDIs\2 Fast 2 Furious.mid.
I0605 18:03:06.882554  8676 convert_dir_to_note_sequences.py:153] Converted MIDI file tmp\\MIDIs\2 Fast 2 Furious.mid.
INFO:tensorflow:Converted MIDI file tmp\\MIDIs\4 Weddings And A Funeral.mid.
I0605 18:03:07.186782  8676 convert_dir_to_note_sequences.py:153] Converted MIDI file tmp\\MIDIs\4 Weddings And A Funeral.mid.
INFO:tensorflow:Converted MIDI file tmp\\MIDIs\9 And Half Weeks.mid.
I0605 18:03:07.660515  8676 convert_dir_to_note_sequences.py:153] Converted MIDI file tmp\\MIDIs\9 And Half Weeks.mid.
INFO:tensorflow:Converted MIDI file tmp\\MIDIs\9 To 5.mid.
I0605 18:03:07.819542  8676 convert_dir_to_note_sequences.py:153] Converted MIDI file tmp\\MIDIs\9 To 5.mid.
```

Figura 3.3: Conversión de 6 MIDIs en Note Sequences

Una vez creado el archivo .tfrecord que contiene la información de los MIDIs en formato *Note Sequences*, el siguiente paso fue dividirlo en *Sequence Examples*, los cuales son dos conjuntos de *Note Sequences*, uno de entrenamiento y otro de validación. Estos forman el conjunto de entradas a las LSTM y representan las melodías de los MIDIs. Para este proceso se utiliza el comando *melody_rnn_create_dataset* con los siguientes argumentos:

- ***–config*:** Después del cual se establece el modelo que se va a utilizar como punto inicial, en este caso es el de *attention_rnn*.
- ***–input*:** Después del cual se establece la ruta y el nombre del archivo con extensión .tfrecord que se ha generado con el comando *convert_dir_to_note_sequences*.
- ***–output_dir*:** Después del cual se establece la ruta donde se quiere que se generen los *Sequence Examples* explicados anteriormente, que serán 2 archivos llamados *eval_melodies.tfrecord* y *training_melodies.tfrecord*.
- ***–eval_ratio*:** Después del cual se establece el ratio de melodías de entrada que se quieren destinar a la validación. En este caso se estableció un 0.15, lo que quiere decir que el 15 % de las BSO de cada género fueron destinadas a validación y el restante 85 % fueron dedicadas para el entrenamiento.

Siguiendo con el ejemplo anterior, en la figura 3.4 se muestra la división del archivo *notesequences.tfrecord* en los 2 conjuntos de *Sequence Examples*. Como se puede observar, de los 6 MIDIs de entrada, 1 es destinado a validación y 5 a entrenamiento, de estos 6 MIDIs se generan realmente 15 entradas en total para las LSTM .

```
$melody_rnn_create_dataset --config=attention_rnn --input=tmp\\notesequences.tfrecord --output_dir=tmp\\sequence_examples --eval_ratio=0.15
2021-06-05 18:16:53.009155: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library cudart64_110.dll
INFO:tensorflow:Processed 6 inputs total. Produced 15 outputs.
I0605 18:17:07.364615 21908 pipeline.py:387] Processed 6 inputs total. Produced 15 outputs.
INFO:tensorflow:DAGPipeline_RandomPartition_eval_melodies_count: 1
INFO:tensorflow:DAGPipeline_RandomPartition_training_melodies_count: 5
```

Figura 3.4: División de Note Sequences a Sequence Examples

Posteriormente, una vez obtenidas las entradas de la RNN , o más en concreto, de la LSTM , lo que quedaba era entrenar la red neuronal. Esto fue llevado a cabo con el comando *melody_rnn_train* con los siguientes parámetros:

- ***–config***: Después del cual se establece el modelo que se va a utilizar como punto inicial del entrenamiento, en este caso es el de *attention_rnn*.
- ***–run_dir***: Después del cual se establece la ruta donde se van a ir guardando los *checkpoints* o puntos de guardado, que almacenan el estado de la LSTM cada x tiempo.
- ***–sequence_example_file***: Después del cual se establece la ruta de los *Sequence Examples* que van a realizar el entrenamiento obtenidos en el paso anterior.
- ***–hparams***: Después del cual se establecen todos los hiperparámetros que se quiere que tenga la RNN . En este caso se configuró una LSTM de dos capas de 64 nodos cada una con una el tamaño del batch o lote de 64. Esta es la configuración recomendada por la web de Magenta [36].
- ***–num_training_steps***: Después del cual se establece el número de pasos en los que se va a llevar a cabo el entrenamiento. Para saber cuántos pasos elegir, se leyeron diferentes investigaciones como [18], de las que se concluyó que un buen número de pasos es 20.000. Esto hizo que los entrenamientos de cada género (son 12) durasen entre 48 y 72 horas, teniendo en cuenta que las NN de los géneros que tenían más número de MIDIs tardaban más. Además, este tiempo depende en gran medida del poder computacional de la máquina donde se esté llevando a cabo el entrenamiento.

En la figura 3.5 se muestra el principio del entrenamiento de los ejemplos anteriores, donde se ve el momento en el que se realizan los *checkpoints* y los diferentes datos del entrenamiento en ese momento como la pérdida, el porcentaje de precisión y la perplejidad (indica el error de forma diferente a la pérdida). En los diferentes géneros se consiguieron los mejores porcentajes de precisión entre los 12000 y 17000 pasos (más o menos a las 36 horas de entrenamiento).

```
$melody_rnn_train --config=attention_rnn --run_dir=tmp\\run1 --sequence_example_file=tmp\\sequence_examples\\training_melodies.tfrecord
                  --hparams="batch_size=64,rnn_layer_sizes=[64,64]" --num_training_steps=20000


2021-06-05 18:56:47.217812: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library cudart64_110.dll
INFO:tensorflow:Calling checkpoint listeners before saving checkpoint 0...
I0605 18:57:03.896984 10428 basic_session_run_hooks.py:613] Calling checkpoint listeners before saving checkpoint 0...
INFO:tensorflow:Saving checkpoints for 0 into tmp\\run1\train\model.ckpt.
I0605 18:57:03.900933 10428 basic_session_run_hooks.py:618] Saving checkpoints for 0 into tmp\\run1\train\model.ckpt.
INFO:tensorflow:Calling checkpoint listeners after saving checkpoint 0...
I0605 18:57:05.607692 10428 basic_session_run_hooks.py:625] Calling checkpoint listeners after saving checkpoint 0...
INFO:tensorflow:Accuracy = 0.022705771, Global Step = 1, Loss = 3.6937263, Perplexity = 40.194344
I0605 18:57:09.926610 10428 basic_session_run_hooks.py:262] Accuracy = 0.022705771, Global Step = 1, Loss = 3.6937263, Perplexity = 40.194344
INFO:tensorflow:Accuracy = 0.68184286, Global Step = 11, Loss = 2.0166302, Perplexity = 7.5129647 (41.226 sec)
I0605 18:57:51.152589 10428 basic_session_run_hooks.py:260] Accuracy = 0.68184286, Global Step = 11, Loss = 2.0166302, Perplexity = 7.5129647 (41.226 sec)
INFO:tensorflow:global_step/sec: 0.0520422
I0605 19:00:22.084292 10428 basic_session_run_hooks.py:702] global_step/sec: 0.0520422
INFO:tensorflow:Calling checkpoint listeners before saving checkpoint 12...
I0605 19:00:26.691145 10428 basic_session_run_hooks.py:613] Calling checkpoint listeners before saving checkpoint 12...
INFO:tensorflow:Saving checkpoints for 12 into tmp\\run1\train\model.ckpt.
I0605 19:00:26.691145 10428 basic_session_run_hooks.py:618] Saving checkpoints for 12 into tmp\\run1\train\model.ckpt.
INFO:tensorflow:Calling checkpoint listeners after saving checkpoint 12...
I0605 19:00:28.601539 10428 basic_session_run_hooks.py:625] Calling checkpoint listeners after saving checkpoint 12...
INFO:tensorflow:Accuracy = 0.6733325, Global Step = 21, Loss = 1.701479, Perplexity = 5.482049 (199.931 sec)
```

Figura 3.5: Entrenamiento de la red neuronal LSTM

3.2.3. Generación de composiciones

El último paso fue, una vez entrenados los modelos de cada género, empezar a generar las melodías monofónicas para cada género. Para ello, se utilizó el comando *melody_rnn_generate*, el cual se ejecuta con los siguientes argumentos:

- ***–config***: Después del cual se establece el modelo que se ha entrenado, en este caso es el de *attention_rnn*.
- ***–run_dir***: Después del cual se establece la ruta donde se han guardado los *checkpoints*.
- ***–output_dir***: Después del cual se establece la ruta donde se quiere que se generen los MIDIs resultantes.
- ***–num_outputs***: Después del cual se establecen el número de MIDIs que se quieren generar.
- ***–num_steps***: Después del cual se establece la duración de la melodía generada. Como referencia, 128 pasos son 8 compases. En este caso, se utilizaron diferentes pasos, entre 64 y 128, ya que menos resultaba muy corta y más empezaba a repetirse mucho la melodía.
- ***–hparams***: Después del cual se establecen los mismos hiperparámetros que se configuraron en el entrenamiento.
- ***–primer_melody***: Después del cual se puede introducir la primera o primeras notas de la melodía que se va a generar. En este caso, se utilizó en todas las composiciones la nota inicial Do, que se corresponde con el número 60.
- ***–primer_midi***: Después del cual se puede introducir la ruta de un MIDI que contenga las notas iniciales de las composición. En este caso, este argumento no se usó ya que se estableció como nota inicial el Do en el argumento *–primer_melody*.

***–qpm*:** Después del cual se establece el número de pulsos por minuto que se van a interpretar los MIDIs resultantes. Por defecto está a 120 y en este caso se dejó este valor.

***–temperature*:** Después del cual se establece la aletoriedad con que se generarán los MIDIs. Por defecto está en 1, y cuanto mayor sea este número, más aleatorio será y viceversa. Se realizaron varias pruebas y al final el valor por defecto es el que arrojaba mejores resultados, por tanto se utilizó este valor.

En la figura 3.6, se puede observar cómo se generan 5 MIDIs cogiendo el último *checkpoint* del entrenamiento, siendo cada uno creado con una *log-likelihood* o función de verosimilitud, la cual indica cómo de bien la melodía generada encaja con el entrenamiento llevado a cabo, por tanto es mejor cuando el valor es mayor.

```
$melody_rnn_generate --config=attention_rnn --run_dir=tmp\\run1 --output_dir=tmp\\MIDIs_finales --num_outputs=5
                     --num_steps=128 --hparams="batch_size=64,rnn_layer_sizes=[64,64]" --primer_melody="[60]"

INFO:tensorflow:Checkpoint used: tmp\\run1\train\model.ckpt-25
INFO:tensorflow:Restoring parameters from tmp\\run1\train\model.ckpt-25
INFO:tensorflow:Beam search yields sequence with log-likelihood: -213.283279
INFO:tensorflow:Beam search yields sequence with log-likelihood: -213.841904
INFO:tensorflow:Beam search yields sequence with log-likelihood: -185.306091
INFO:tensorflow:Beam search yields sequence with log-likelihood: -181.601608
INFO:tensorflow:Beam search yields sequence with log-likelihood: -187.022919
INFO:tensorflow:Wrote 5 MIDI files to tmp\\MIDIs_finales
```

Figura 3.6: Generación de nuevas composiciones musicales en formato MIDI

3.2.4. Comprobación de los resultados obtenidos

Una vez generados unos 40 MIDIs para cada género, se eligieron los mejores de cada uno, teniendo en cuenta si el sonido era melódico. Para verificar que esos MIDIs elegidos fuesen lo suficientemente buenos como para pensar que un humano lo hubiese podido componer, se llevó a cabo el Test de Turing (TT) .

Este método fue propuesto por Alan Turing y consiste en una prueba realizada a personas para ver si son capaces de distinguir si algo está hecho por una máquina o por un humano.

En este trabajo se ha utilizado el TT para comprobar si un grupo de personas es capaz de distinguir si una composición musical está hecha por una persona o por la RNN explicada en esta sección. En la sección 4.3 se detallará este experimento.

3.3. Clasificación de bandas sonoras

Esta sección realmente se trata de cómo se llevó a cabo la investigación sobre si realmente las BSO se pueden clasificar en los diferentes géneros cinematográficos al igual que las películas en las que aparecen o, por el contrario, la música de una película no depende en absoluto de su género.

3.3.1. Herramientas utilizadas

Para llevar a cabo esta investigación se ha buscado sobre la mejor forma de clasificar música, y tal y como se detalló en 2.1.6, esta es con CNN . Sin embargo, las CNN procesan imágenes y no sonido, pero como también se detalló en esa subsección 2.1.6, el sonido se puede convertir a imagen, y en este caso, se convierte en los llamados espectrogramas. Esta conversión es llevada a cabo con las librerías de librosa [38] y matplotlib [39] (está en **conversion_MP3_PNG.py** del GitHub de este TFG [27]). Un ejemplo de un archivo MP3 convertido en espectrograma utilizado en este estudio se muestra en la figura 3.7.

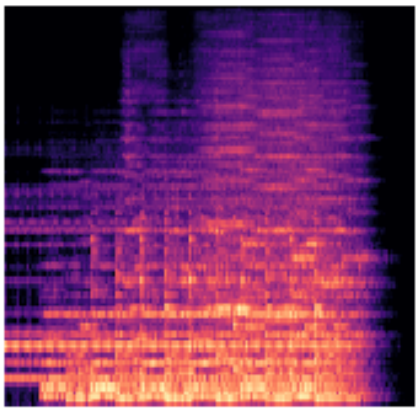


Figura 3.7: Ejemplo de espectrograma utilizado

Para el código de las CNN , se utilizaron las librerías de Tensorflow [40], Keras (librería que pertenece a Tensorflow) y Scikit-learn [41]. Estas librerías son muy utilizadas en el ámbito del ML .

Aparte de investigar esto con CNN , también se estudió si los humanos son capaces de diferenciar a qué género cinematográfico corresponden las BSO , con un formulario de Google que se explicará en detalle en el capítulo 4.

3.3.2. Desarrollo de las CNN

Para todas las CNN desarrolladas en los experimentos (secciones 4.1 y 4.2), el primer paso fue construir un *Dataframe* con Pandas [42]. Un *Dataframe* consiste en una estructura de datos en dos dimensiones, en este caso, estaba compuesto por los nombres de las imágenes de los espectrogramas relacionados con los géneros cinematográficos de las películas a las que pertenecen estas imágenes.

Una vez creado el *Dataframe*, se dividió en 2 partes: entrenamiento y test. Esta división es hecha de manera aleatoria con la semilla que se quiera.

El siguiente paso fue generar más imágenes de entrenamiento a partir de las originales, es decir, generar imágenes, las cuales siguen relacionadas a los mismos géneros cinematográficos que las originales, aplicándolas distintos zooms y volteándolas horizontalmente. Además, todas estas imágenes, tanto las nuevas como las originales y tanto las de entrenamiento como las de test, fueron normalizadas para aumentar la precisión en las CNN posteriores.

Los últimos pasos antes de establecer las capas de la CNN son extraer el contenido de las imágenes para meterlo en la red, ya que antes solo se tenía su nombre pero no su contenido, establecer el tamaño que tienen estas imágenes y elegir cómo van a ser las salidas estas CNN .

Respecto a las capas establecidas en los diferentes experimentos, se van a aplicar principalmente las capas de convolución y *MaxPooling* explicadas en 2.1.5.

En la compilación, se ha utilizado el optimizador *Adam* (que adapta un ratio de aprendizaje en función de cómo estén distribuidos los parámetros de la red), la función de pérdida *binary corssentropy* y la métrica *accuracy*. Lo siguiente a la compilación, es el entrenamiento llevado a cabo con diferentes épocas en los distintos experimentos.

Finalmente, se creó un gráfico por cada entrenamiento llevado a cabo, donde se puede apreciar la *accuracy* o precisión de cada época tanto en el entrenamiento como en la validación.

4

Experimentos y resultados

En este capítulo se van a explicar los diferentes experimentos llevados a cabo tanto para saber si realmente es viable clasificar las BSO en géneros cinematográficos como para ver si las melodías generadas en este trabajo son capaces de pasar por composiciones realizadas por humanos.

4.1. Experimento 1: CNN para todos los géneros

El primer experimento llevado a cabo fue configurar una CNN para ver si esta era capaz de clasificar el género de las BSO . Esta fue alimentada por los datos extraídos en el segundo estudio de la sección 3.1 (565 MP3) y configurada como se indica en la sección 3.3 y a continuación. El código de este experimento está en el archivo **experimento1.py** del GitHub de este trabajo [27].

En principio, parecían pocos archivos, pero teniendo en cuenta que se generaban más imágenes para el entrenamiento con diferentes zooms y volteos horizontales, se decidió probar.

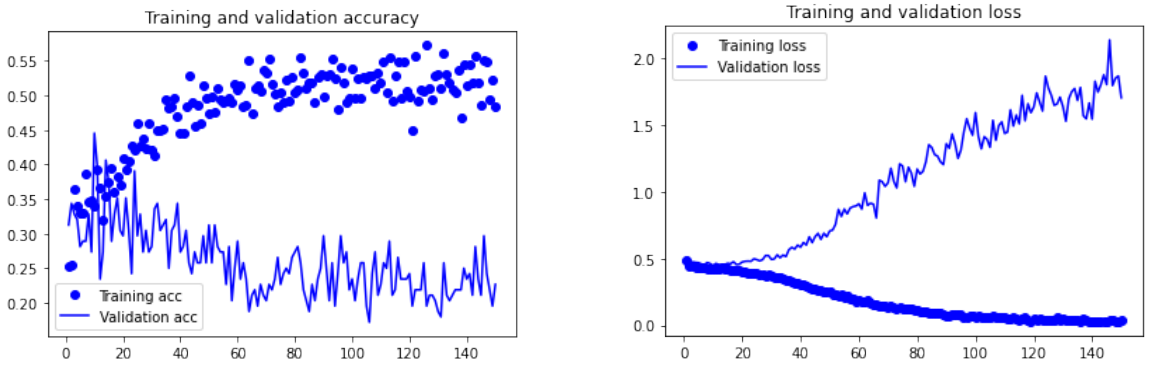
En este caso se consideraron BSO de todos los géneros seleccionados (12), lo que significa que una pieza musical podía pertenecer a uno o **más** géneros cinematográficos. Por ello, se tuvo que hacer *One-Hot Encoding*, que consiste en convertir el *Dataframe* inicial en otro que es una especie de tabla, la cual se basa en poner un 0 si no pertenece al género correspondiente o un 1 si sí pertenece. El resultado de esta conversión está mostrado en la figura B.1.

Una vez hecho el *One-Hot Encoding* y siguiendo los pasos descritos en 3.3, se dividieron los datos de entrada en entrenamiento y en test, con un porcentaje de 75 %-25 % respectivamente. La elección de cómo tenían que ser las salidas, siendo coherente con que cada banda sonora puede pertenecer a uno o más géneros, fue de tipo *raw*, ya que al introducir una nueva pieza musical en la red, da como salida porcentajes de parecido a cada género existente, cogiendo como géneros predichos los que sean mayores o iguales al 50 %.

En cuanto a la composición de este experimento, se probaron muchas combinaciones. Sin embargo, ningún resultado fue especialmente bueno, de hecho todo lo contrario. La composición que dio mejor resultado fue utilizar 3 capas de convoluciones, todas ellas con un kernel de 3x3, con la función

de activación *ReLU* y con 32, 64 y 128 filtros respectivamente. Estás capas de convoluciones están separadas por 3 capas iguales de *Max Pooling*, con un tamaño de pool de 2x2. Después de estas capas, se ponen todos los filtros en una sola dimensión y se forma una capa *fully connected*, de 256 neuronas y con una función de activación *ReLU*, que a su vez está conectada a otra capa tradicional de 512 neuronas y con la misma función de activación. A esto se añade una capa de *Dropout*, que se encarga de normalizar las salidas. Por último, se añade una capa con la función de activación *sigmoide* que cuenta con 12 neuronas, una por cada género cinematográfico.

En la figura 4.1, se muestran la precisión y la pérdida tras un entrenamiento de 150 épocas, indicando tanto lo obtenido en el entrenamiento como en la validación. Se puede observar que la máxima *Training accuracy* llega más o menos hasta un 0,55, una cifra baja. Además, se ve que en la validación el máximo es de un 0,45, el cual solo ocurre al principio cuando esto es casi aleatorio, de hecho tiende a estabilizarse en 0,25, una cifra muy baja con la que se puede concluir que no ha conseguido aprender a diferenciar géneros esta CNN . En el principio se visualiza un proceso conocido como *overfitting* o sobreajuste, en el cual el entrenamiento aprende a reconocer patrones muy determinados que nada tienen que ver con el género cinematográfico, pero que se cumplen en algunos datos del conjunto de entrenamiento, por eso es capaz de subir hasta un 0,55 y, sin embargo, en la validación ocurre el proceso contrario, ya que está identificando patrones incorrectos. Esto se ha intentado paliar de diferentes formas, como añadiendo capas de *Dropout* o haciendo más sencilla la red, sin embargo, se remarca que los mejores resultados conseguidos son con esta composición.



(a) Training and validation accuracy

(b) Training and validation loss

Figura 4.1: Resultados del primer experimento

Para poder determinar qué es lo que estaba fallando se creó una matriz de confusión (mostrada en el apéndice C), la cual es una matriz que compara el género cinematográfico al que realmente pertenece una banda sonora con el predicho por la CNN. En ella se puede observar cómo predice algo mejor los géneros que tienen muchas más BSO. Esto también se puede observar en la siguiente figura 4.2, que muestra, para cada género, el porcentaje de aciertos, el número de BSO utilizadas para la validación y el número total de estas.

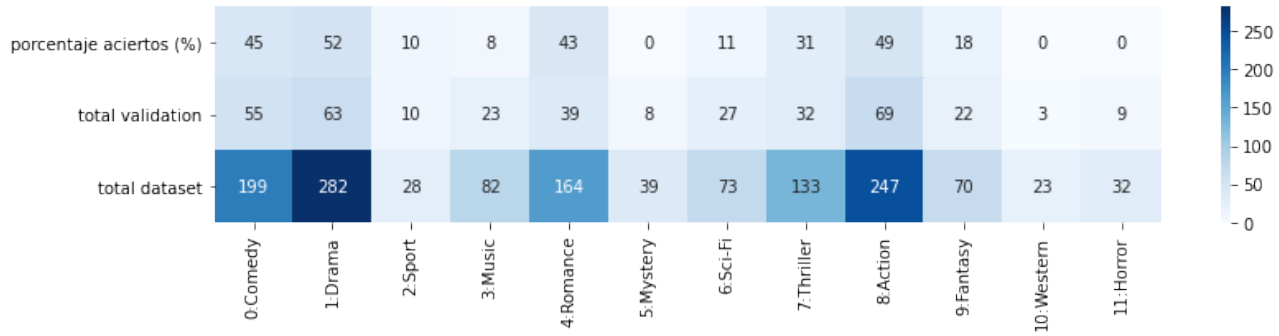


Figura 4.2: Tabla resumen del primer experimento

Como conclusión de este experimento se determinó que las posibles razones del mal funcionamiento de esta CNN pueden ser el número de muestras usadas (565), que los géneros no tengan similar número de BSO o que esta red es demasiado compleja, ya que además de tener que decidir entre 12 géneros, tiene que predecir si una banda sonora pertenece a uno o varios de esos géneros. De estas conclusiones surge el segundo experimento.

4.2. Experimento 2: CNN para cada género

Este segundo experimento fue pensado para intentar mejorar la precisión en la clasificación y resolver los problemas surgidos en el entrenamiento del experimento anterior. El código de este experimento está en el archivo **experimento2.py** del GitHub de este trabajo [27].

Para este segundo experimento, se utilizaron los datos extraídos del tercer estudio de la sección 3.3 (alrededor de 60000 MP3) y la CNN utilizada está compuesta de forma similar a la del primer experimento pero con algunos cambios detallados a continuación.

La diferencia más importante es que no consiste en una única CNN , sino que para que este experimento prediga el o los géneros cinematográficos de una banda sonora, tiene que pasar por 12 CNN , una por cada género. Esto significa que la salida de cada CNN consiste en decir si pertenece o no al género que es capaz de predecir esa red neuronal. La explicación se centra en la CNN de comedia, que intentará predecir si una banda sonora pertenece al género de comedia o no, para todas las demás redes de cada género el procedimiento es el mismo.

Otra diferencia en la implementación es que se van a coger el mismo número de muestras que pertenezcan al género cinematográfico que las muestras que no pertenezcan a ese género. Esto se hace para que, como pudo haber ocurrido en el experimento anterior, no prediga sin indicios que pertenece al género que posee un mayor número de muestras, ya que entonces los resultados no serían completamente fiables.

Por tanto, para esta implementación, se hace un recuento y se cogen las piezas musicales que pertenecen a Comedia y también se cogen ese mismo número de BSO pero que no pertenecen a Comedia de forma aleatoria.

En este caso, no hubo que hacer *One-Hot Encoding* ya que solo pueden clasificarse en *Comedy* o *non-Comedy* (el *Dataframe* está representado en la figura B.2), refiriéndose a si pertenece al género de la CNN o no. Por ello, la elección de cómo tenían que ser la salidas fue de tipo *binary*.

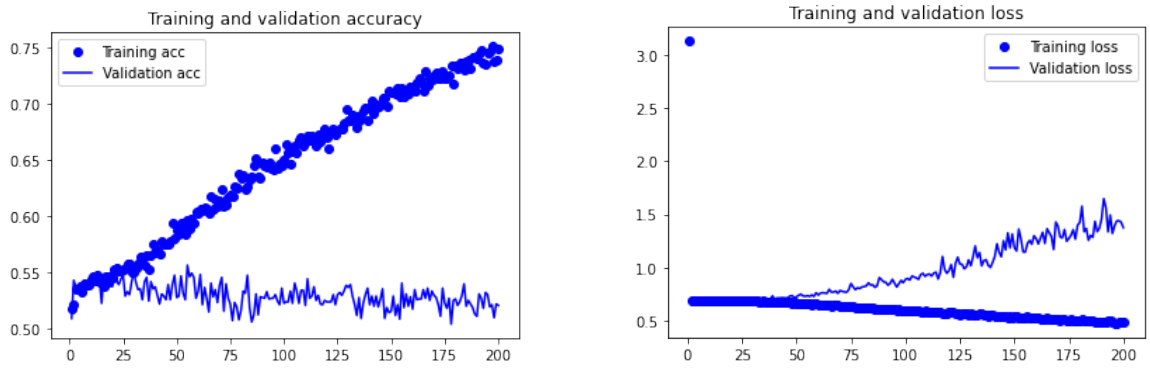
En cuanto a la composición de las CNN , se han probado diversas combinaciones de capas y parámetros en un estudio llevado a cabo durante un tiempo significativo debido a que cada prueba de 100 épocas tarda alrededor de 48 horas por sus amplias muestras. Las CNN están compuestas por una sola capa de convolución de 128 filtros, con función de activación *ReLU* y un kernel de 3x3, solo hay una debido a que una complejidad mayor en una red neuronal da lugar a un incremento del *overfitting*. Esta capa va unida a su correspondiente capa de *Max Pooling*, la cual tiene un tamaño de pool de 2x2. Posteriormente, se ha añadido una capa de Dropout para normalizar y también intentar paliar el Overfitting. Después, con el *Flatten* se aplanan todos los filtros y se conecta a la capa *fully connected* de 256 neuronas y con una función de activación *ReLU*. Además de solo haber una y no dos como en el experimento anterior, se añade otra capa de *Dropout* para seguir reduciendo el *Overfitting*. Por último, se conecta a una capa con la función de activación *sigmoide* de una neurona, la cual decide si es o no es de ese género cinematográfico.

Como se puede observar en la figura 4.3, tras un entrenamiento de 200 épocas, la *training accuracy* no se estabiliza y podría seguir aumentando con más épocas, sin embargo, lo realmente importante que es que la red neuronal aprenda para los datos nuevos, es decir, que la *validation accuracy* aumente, esto no ocurre ya que se queda estancada en un 0,55 desde el principio. Esto significa que sigue habiendo *Overfitting* a pesar de todas las medidas llevadas a cabo para evitarlo como las capas puestas de *Dropout* y como hacerla lo más simple posible. Se han hecho pruebas aumentando estas capas de normalización, sin embargo, añadir más significaba que ni el entrenamiento era capaz de aprender, aparte no se podía simplificar más la arquitectura de esta red.

Dado que la clasificación por géneros con redes neuronales no da los resultados deseados debido al *Overfitting*, se plantea un tercer experimento para que sean las propias personas las que clasifiquen las melodías en estos géneros.

4.3. Experimento 3: Cuestionario

Este experimento tiene 2 objetivos: Determinar si las personas son capaces de diferenciar el género cinematográfico escuchando melodías de BSO y determinar si las composiciones generadas en la sección 3.2 son capaces de pasar el Test de Turing.



(a) Training and validation accuracy

(b) Training and validation loss

Figura 4.3: Resultados del segundo experimento

Para ello, se han seleccionado 5 melodías monofónicas de 5 géneros cinematográficos distintos (Comedia, Drama, Ciencia Ficción, Fantasía y Acción) generadas en la sección 3.2. Además, se han seleccionado otras 5 melodías monofónicas de bandas sonoras reales (poco conocidas), siendo cada una de uno de los 5 géneros escogidos. Estas melodías son expuestas en el Apéndice D.

Con estas piezas musicales, se ha elaborado un cuestionario en Google Forms, al que se puede acceder a través del siguiente enlace: `https://forms.gle/17ETjLyUowe1J9cB7`. En este formulario se pregunta el nivel musical y cinematográfico por si se pudiesen sacar conclusiones en base a los conocimientos previos. Este consta de 5 apartados, en cada uno de los cuales se encuentra un enlace a YouTube de un vídeo en el que se reproducen y se muestran dos piezas musicales (una real y otra artificial). Después de cada uno de esos vídeos, se pregunta al encuestado que intente reconocer cuál de las 2 composiciones ha sido compuesta por una máquina. Acto seguido se pregunta por el género cinematográfico al que creen que pertenece cada una de las 2 piezas musicales pero dando solo 3 opciones.

Se han obtenido 201 respuestas hasta la realización de esta memoria, las cuales se detallan en profundidad en el Apéndice D. En la figura 4.4, se indican los porcentajes de aciertos y de fallos que se han dado en la parte de elegir la composición artificial, siendo filtrados para cada nivel musical (siendo 0 el menor y 5 el mayor) y mostrándose la media total de todos los niveles. Como se puede observar, hay una clara confusión sobre cuáles son las composiciones artificiales y las reales. Otro dato interesante, es que las personas con mayor nivel musical (este cuestionario ha sido distribuido en 10 conservatorios), son las que más se han confundido, significando esto que ven a las composiciones artificiales una pieza con mucho sentido musical.

Por otra parte, en las figuras 4.5, se indican los porcentajes de acierto que han tenido las personas a la hora de clasificar las composiciones, tanto artificiales como reales, según el género cinematográfico. Como se puede observar en la 4.5(a), el porcentaje medio de acierto está en torno al 44 % mientras que

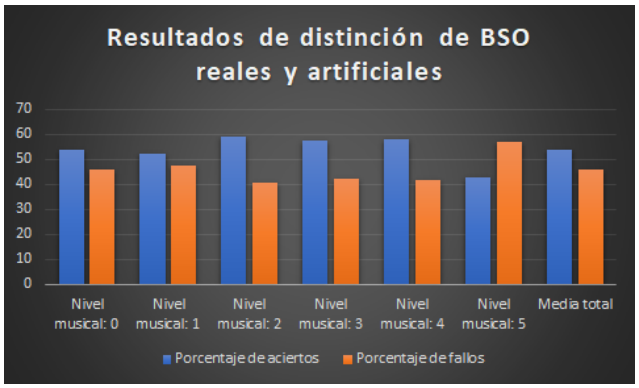
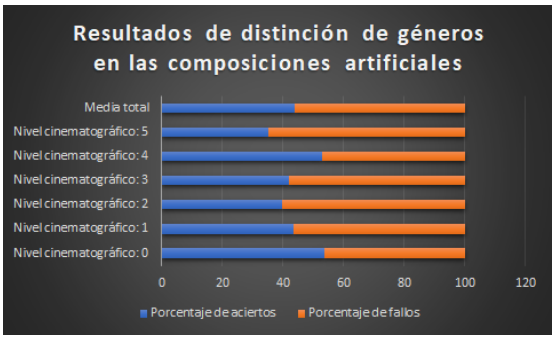
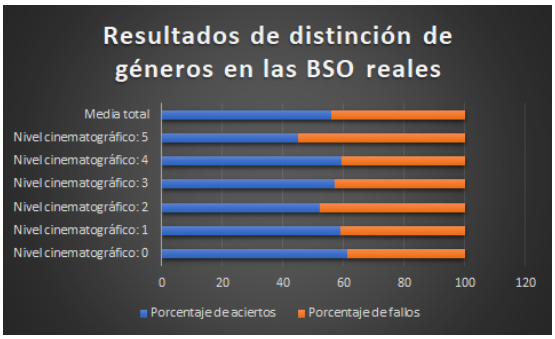


Figura 4.4: Resultados de distinción de BSO reales y artificiales

en la figura 4.5(b), este porcentaje es de un 56 %. Al haber 3 opciones, la predicción aleatoria tendría, idóneamente, un porcentaje de acierto de un 33 %. Todo esto concluye que clasificar el género parece una difícil tarea, a pesar de haber elegido las melodías más acordes a estos géneros seleccionados. Respecto al nivel cinematográfico, no hay mucho que decir, ya que parece que importa poco, de hecho, las personas menos cinéfilas (0), son las que más aciertan, lo cual hace ver que el conocimiento de películas poco tiene que ver a la hora de clasificar una banda sonora por su género cinematográfico.



(a) En composiciones artificiales



(b) En composiciones reales

Figura 4.5: Resultados de distinción de géneros

5

Conclusiones y trabajo futuro

5.1. Conclusiones

Con la tecnología actual, hoy en día, es posible la creación de música de manera artificial, pudiendo establecer de qué tipo, género o incluso artista se quiere componer, siempre y cuando se tengan composiciones previas con las que estas máquinas puedan aprender. Sin embargo, de momento, esto tiene sus limitaciones.

En este trabajo se ha expuesto la creación de melodías monofónicas para películas, a partir del género cinematográfico. Estas composiciones son capaces de pasar el Test de Turing, es decir, muchas personas no pueden distinguir si están compuestas por una máquina o por otro ser humano, pero esto solo ocurre si todas las melodías con las que se comparan tienen las mismas características musicales. Este buen resultado se debe principalmente a que estas composiciones artificiales están formadas por notas musicales melódicas que poseen una duración coherente, lo cual genera una melodía con sentido. El problema de las composiciones generadas en este trabajo son la duración, es decir, no pueden ser más largas de 30 segundos ya que empiezan a ser demasiado repetitivas; solo están interpretadas por un instrumento, el cual es un Gran Piano Acústico; son monofónicas; y no se genera ningún matiz musical, es decir, todas las partes de la melodía suenan con la misma intensidad. Seguramente estas piezas musicales artificiales no encajarían tal cual son en escenas de películas, sin embargo, estas podrían servir como referencia de melodías principales o secundarias en una banda sonora, las cuales además se podrían embellecer con la inclusión de más instrumentos.

Respecto a la gran pregunta formulada en este trabajo de *¿sería posible clasificar las bandas sonoras en función del género cinematográfico de la película en la que aparecen?* En la investigación llevada a cabo, se puede concluir en que no es posible diferenciarlas, al menos de una forma tajante, ni por personas ni por máquinas.

Es cierto que hay algunas melodías que hacen sentir terror, tristeza y otras muchas emociones. De hecho, una película está repleta de bandas sonoras que reflejan estas emociones, que además son interpretadas visualmente en cada una de las escenas, pero esas emociones pueden variar en una misma película, por tanto clasificar según el género cinematográfico es algo demasiado general, que a veces, no se corresponde directamente con la realidad. Por ejemplo, en una película de Acción puede haber escenas tristes, las cuales son reflejadas en la música de fondo, sin embargo, esa película no es clasificada como Drama sino como Acción.

Una vez presentada esta conclusión, puede parecer extraño el hecho de que las bandas sonoras generadas en este trabajo se crean en base a un género cinematográfico. Sin embargo, lo que se quiere reflejar realmente, es que las máquinas son capaces de crear cualquier tipo de música siempre y cuando tengan muestras de las que aprender, independientemente del uso final que se vaya a dar.

5.2. Trabajo futuro

A partir de este trabajo surgen nuevas líneas de investigación:

1. **Mejorar la generación de bandas sonoras:** Como ya se ha visto, se han podido generar bandas sonoras pero con algunas limitaciones. Por ello, se propone investigar sobre la generación de melodías armónicas (más de una línea melódica), interpretadas por varios instrumentos y de la duración que se quiera.
2. **Investigar la clasificación de bandas sonoras según los sentimientos producidos:** En este trabajo, se ha investigado la posibilidad de clasificar las bandas sonoras según el género cinematográfico de las películas en las que aparecen, sin embargo, la conclusión es que no se puede. Por tanto, el objetivo sería investigar si estas bandas sonoras, normalmente asociadas a escenas de las películas, se pudiesen clasificar según los sentimientos que producen estas. Para ello, se podrían utilizar las CNN desarrolladas en este TFG o unas similares. Sin embargo, esto requeriría un etiquetado difícil de encontrar o, si es hecho a mano, sería un trabajo muy laborioso. De cualquier forma sería muy interesante e incluso se podría a llegar a combinar con el anterior trabajo futuro propuesto, con la finalidad de generar bandas sonoras basadas en el sentimiento que producen.
3. **Generar melodías basadas en vídeos:** Por último, se propone una variante en la generación de música, crear composiciones musicales para cualquier tipo de vídeo, ya sea un vídeo de Youtube, un anuncio, una escena de una película... Esto podría tener un amplio uso, sobre todo en el tema de evitar denuncias por usar música de otras personas (Copyright), ya que se estarían utilizando melodías generadas por uno mismo, introduciendo el vídeo deseado en la tecnología desarrollada.

Bibliografía

[1] N. P. G. y Marcela Sánchez, "Inteligencia Artificial: La Cuarta Revolución Industrial," *ayming*, 2019. (Acceder).

[2] L. Rouhiainen, *Inteligencia artificial*. Editorial Planeta, 2018. (Acceder).

[3] I. R. Ozcariz, "Segmentación semántica multiclase de imágenessubmarinas utilizando redes neuronales profundas," Master's thesis, Universitat de les Illes Balears, 2019. (Acceder).

[4] O. Sanseviero, "AI en 3 minutos: Tipos de Machine Learning," *medium.com*, 2018. (Acceder).

[5] J. I. Bagnato, *Aprende Machine Learning en Español*. Agencia del ISBN, 2020. (Acceder).

[6] N. E. Estapé, "¿Cuál es la diferencia entre el machine learning y el deep learning?," *bismart*, 2020. (Acceder).

[7] V. Vásquez, "DEL HELIOCENTRISMO GRIEGO A LA REVOLUCIÓN COPERNICANA," *metodo2013.blogspot.com*, 2014. (Acceder).

[8] J. D. V. García, "Redes neuronales desde cero (I) – Introducción," *IArtificial.net*, 2020. (Acceder).

[9] M. Dozmorov, "Day 2: Fundamentals of Deep Learning II," *bios691-deep-learning-r.netlify.app*, 2020. (Acceder).

[10] R. Mendoza, "Entendiendo las Redes Neuronales Artificiales," *medium*, 2019. (Acceder).

[11] Anonymous, "Understanding single layer Perceptron and difference between Single Layer vs Multilayer Perceptron," *i2tutorials*, 2019. (Acceder).

[12] D. Calvo, "Clasificación de redes neuronales artificiales," *diegocalvo.es*, 2017. (Acceder).

[13] J. Barrios, "Redes Neuronales Convolucionales," *juanbarrios.com*, 2020. (Acceder).

[14] C. Kuliah, "Student Notes: Convolutional Neural Networks (CNN) Introduction," *indoml*, 2018. (Acceder).

[15] A. G. Walters, "Convolutional Neural Networks (CNN) to Classify Sentences," *austingwalters.com*, 2019. (Acceder).

[16] A. Mittal, "Understanding RNN and LSTM," *Medium*, 2019. (Acceder).

[17] C. Olah, "Understanding LSTM Networks," *colah.github.io*, 2015. (Acceder).

[18] S. J. Gómez, "Generación y evaluación de secuencias melódicas mediante Inteligencia Artificial," Master's thesis, Universidad Politécnica de Madrid, 2019. (Acceder).

[19] A. Chowdhry, "Music genre classification using cnn," *clairvoyantsoft*, 2021. (Acceder).

[20] N. Singh, "Identifying the genre of a song with neural networks," *medium*, 2018. (Acceder).

[21] I. Martinez, "La musica," *calameo*, 2017. (Acceder).

[22] M. E. Raffino, "Concepto de musica," *concepto.de*, 2020. (Acceder).

[23] Anónimo, "Introducción a la escala," *ciudadpentagrama.com*, 2018. (Acceder).

[24] B. Kraemer, "Símbolos musicales de la música para piano: Segunda parte," *aboutespanol*, 2019. (Acceder).

[25] O. S. Hiremath, “A beginner’s guide to learn web scraping with python!,” *edureka*, 2020. (Acceder).

[26] G. Simões, J. Wehrmann, R. Barros, and D. Ruiz, “Movie genre classification with convolutional neural networks,” *researchgate*, pp. 259–266, 07 2016.

[27] A. S. Abad, “Tfg: Estudio de la composición de bandas sonoras para cine mediante inteligencia artificial,” 2021. (Acceder).

[28] V. Autores, “themoviedb.” (Acceder).

[29] V. Autores, “freemidi.” (Acceder).

[30] K. Reitz, “Requests: Http para humanos.” (Acceder).

[31] L. Richardson, “beautifulsoup4 4.9.3.” (Acceder).

[32] V. Autores, “imdb.” (Acceder).

[33] C. Concept and Creation, “soundtrackcollector.” (Acceder).

[34] D. sigma67, “ytmusicapi: Unofficial api for youtube music.” (Acceder).

[35] S. M. Remita Amine, “youtube-dl.” (Acceder).

[36] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, “A hierarchical latent vector model for learning long-term structure in music,” in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 4364–4373, PMLR, 10–15 Jul 2018.

[37] V. Autores, “Magenta repositorio github.” (Acceder).

[38] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, “librosa: Audio and music signal analysis in python,” in *Proceedings of the 14th python in science conference*, vol. 8, 2015.

[39] J. D. Hunter, “Matplotlib: A 2d graphics environment,” *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[40] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. Software available from tensorflow.org.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] Wes McKinney, “Data Structures for Statistical Computing in Python,” in *Proceedings of the 9th Python in Science Conference* (Stéfan van der Walt and Jarrod Millman, eds.), pp. 56 – 61, 2010.

Definiciones

alteración En la música, una alteración aplicada a una nota musical consiste en modificar su altura. Las alteraciones que existen son el bemol, el sostenido y el becuadro.

clase En el contexto de la clasificación por Inteligencia Artificial, hace referencia a las predicciones hechas por la máquina; por ejemplo, si se está utilizando aprendizaje automático para predecir si una imagen es un perro o un gato, las dos clases que existen son perro y gato.

cookie Es información creada por las páginas web con diferentes fines que se almacena en el cliente, es decir, en el navegador web de quien está accediendo a esa página web.

HTML Es el lenguaje que se utiliza para la elaboración de páginas web. HTML son las siglas de HyperText Markup Language.

monofónica Es la melodía más sencilla que se puede dar en la música, consiste en una única línea melódica sin acompañamiento ninguno.

partitura Es la representación escrita de una obra musical.

URL Sus siglas significan Uniform Resource Locator y es una dirección que apunta a un recurso en la web.

Acrónimos

API Application Programming Interfaces.

BSO Bandas Sonoras.

CNN Convolutional Neural Network.

DL Deep Learning.

I1 Investigación 1.

I2 Investigación 2.

I3 Investigación 3.

IA Inteligencia Artificial.

LSTM Long Short Term Memory.

MIDI Musical Instrument Digital Interface.

ML Machine Learning.

NN Neural Networks.

RGB Red, Green and Blue.

RL Reinforcement Learning.

RNN Recurrent Neural Network.

TFG Trabajo de Fin de Grado.

TT Test de Turing.

WS Web Scraping.

Apéndices

A

Flujograma de la segunda extracción de datos

En esta apéndice se muestra la secuencia de pasos realizados en la extracción de datos del segundo estudio en forma de flujograma.

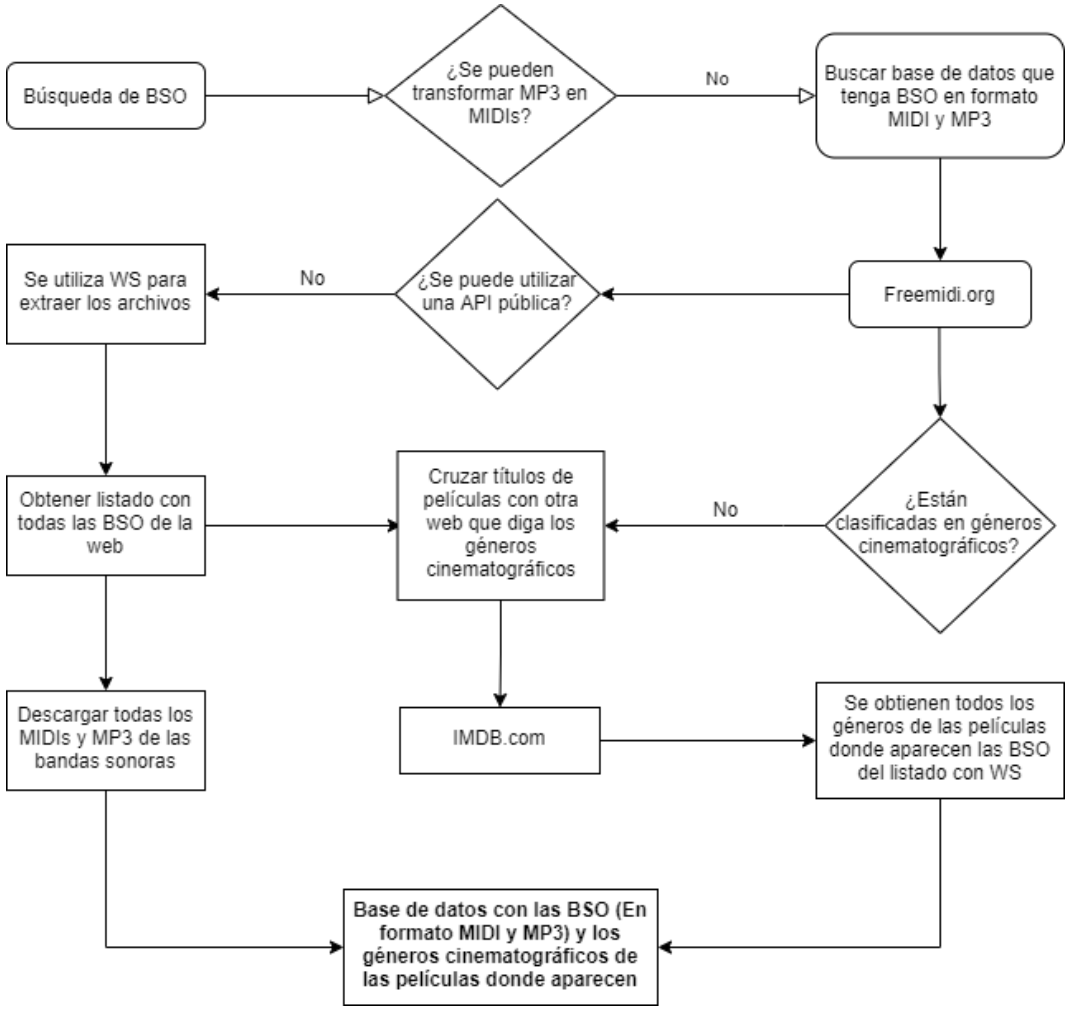


Figura A.1: Flujograma de la extracción de datos en el segundo estudio

B

Dataframes introducidos en las CNN

En este apéndice se muestran los Dataframes que se introducen en las CNN de los experimentos 1 y 2.

B.1. Experimento 1

	filename	Comedy	Drama	Sport	Music	Romance	Mystery	Sci-Fi	Thriller	Action	Fantasy	Western	Horror
0	10.png	1	0	0	0	1	0	0	0	0	0	0	0
1	101 Dalmations.png	1	0	0	0	0	0	0	1	1	0	0	0
2	12 Monkeys.png	0	0	0	0	0	1	1	1	0	0	0	0
3	1492 Conquest of Paradise.png	0	1	0	0	0	0	0	0	1	0	0	0
4	1941.png	1	0	0	0	0	0	0	0	1	0	0	0
..	...	...	...	...	...	...	...	...	...	...	...	...	...
560	Yellow Submarine.png	1	0	0	1	0	0	0	0	1	1	0	0
561	You've Got Mail.png	1	1	0	0	1	0	0	0	0	0	0	0
562	Young Guns.png	0	0	0	0	0	0	0	0	1	0	1	0
563	Yours Mine And Ours.png	1	0	0	0	0	0	0	0	0	0	0	0
564	Zorba The Greek.png	1	1	0	0	0	0	0	0	0	0	0	0

[565 rows x 13 columns]

Figura B.1: Dataframe del experimento 1

B.2. Experimento 2

	filename	category
0	!Jo, papa! (1975).png	Comedy
1	'A' gai wak 2.png	Comedy
2	'Imuhar', une legende (1997).png	non-Comedy
3	'night, Mother.png	non-Comedy
4	...All the Marbles.png	Comedy
...	...	...
7926	Zikina dinastija (1985).png	Comedy
7927	Zivot je cudo.png	Comedy
7928	Zombi 2.png	non-Comedy
7929	Zui quan.png	Comedy
7930	Zycie za zycie. Maksymilian Kolbe.png	non-Comedy

[7931 rows x 2 columns]

Figura B.2: Dataframe del experimento 2

C

Matriz de confusión del primer experimento

En este apéndice se muestra la matriz de confusión resultante del experimento 1:

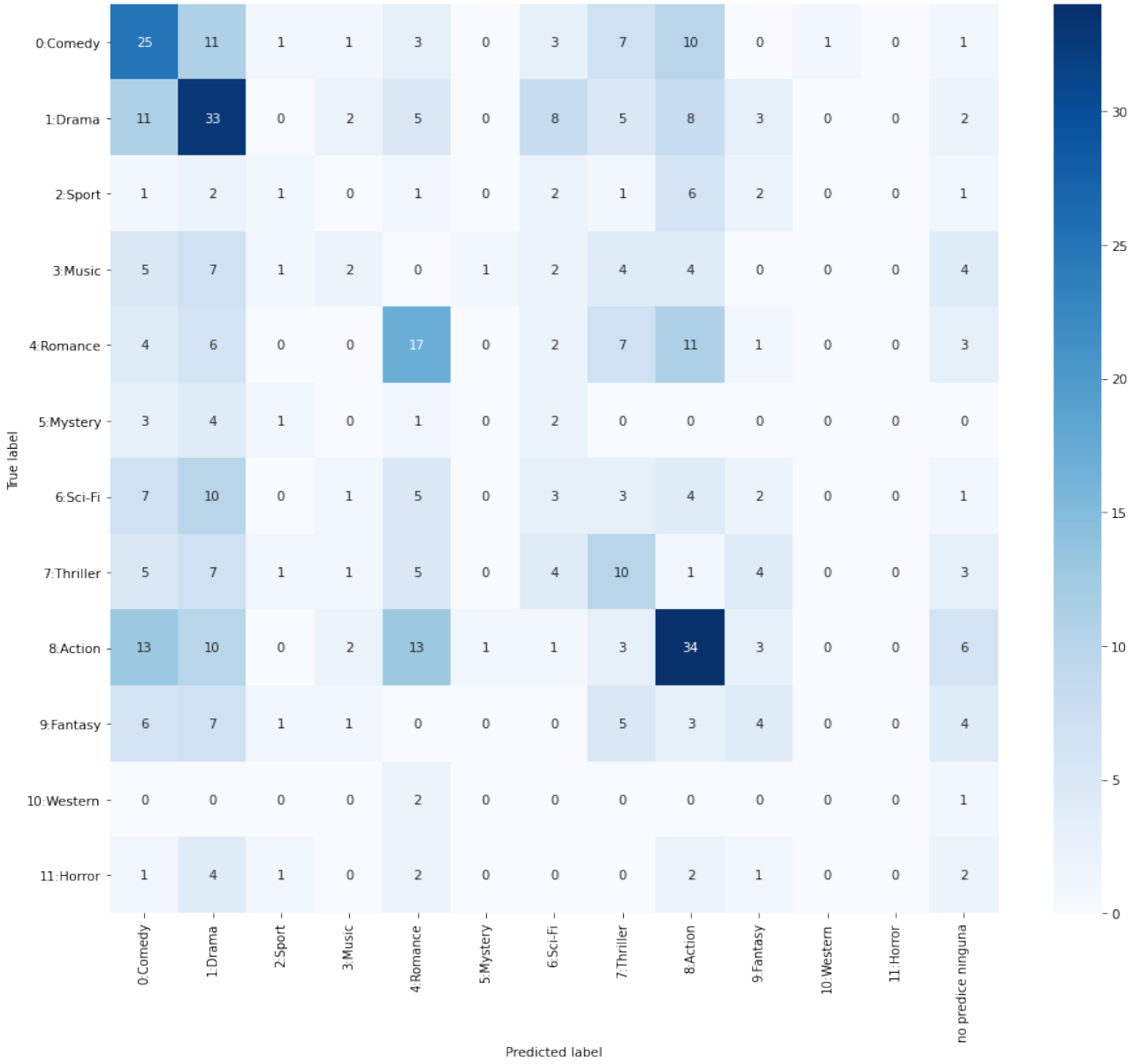


Figura C.1: Matriz de confusión primer del primer experimento

D

Cuestionario

En este apéndice se detallan, por cada apartado del cuestionario, las composiciones sobre las que se han realizado las preguntas y los resultados que se han obtenido en cada una de ellas. El enlace a este cuestionario es el siguiente: `https://forms.gle/17ETjLyUowe1J9cB7`. Cada apartado contiene dos composiciones monofónicas, una artificial y otra extraída de una banda sonora real. Como apunte, en las preguntas de elegir el género de cada composición, hay respuestas con más de 3 géneros ya que en una primera versión (para 3 personas) se dieron 5 opciones, pero como se vio que el porcentaje de aciertos era muy bajo, se acotaron a 3 opciones.

D.1. Nivel musical y cinematográfico

Figura D.1: Nivel musical

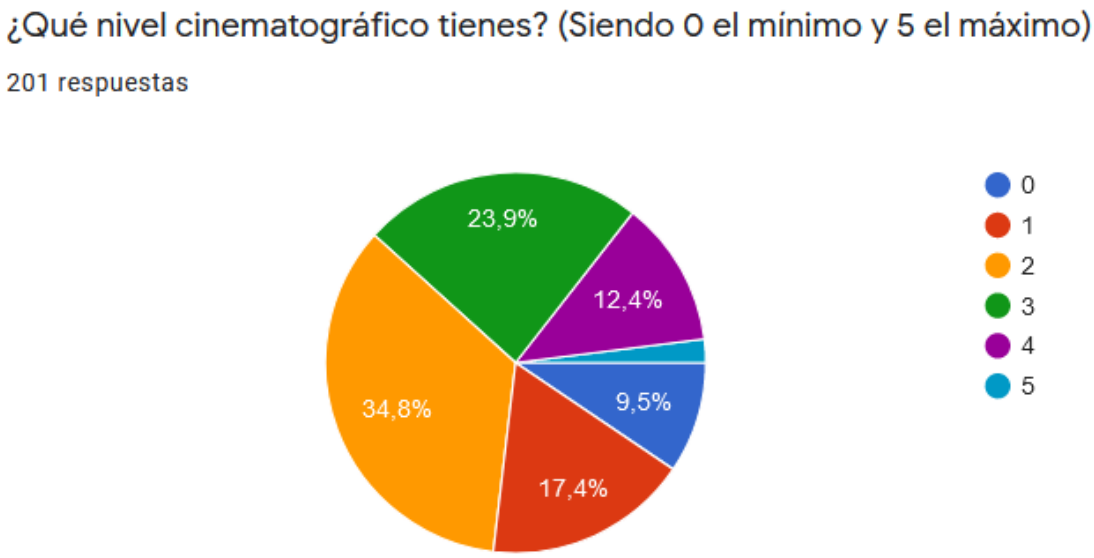


Figura D.2: Nivel cinematográfico

D.2. Apartado 1

En el siguiente enlace se reproducen las dos primeras composiciones: `https://youtu.be/xTK0EDJXzMo`.

La composición artificial está mostrada en la figura D.3 y pertenece a Ciencia Ficción.

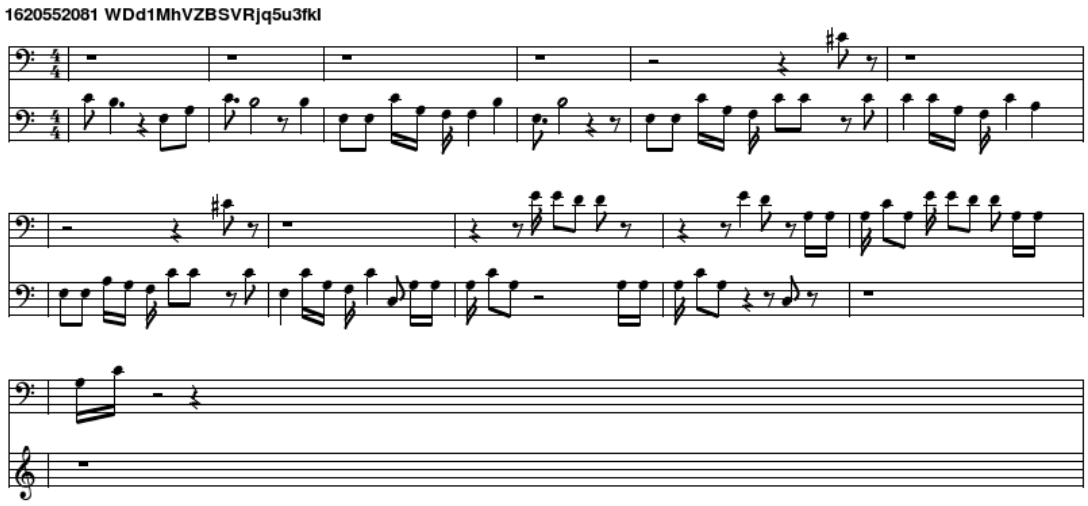


Figura D.3: Primera composición artificial

La composición extraída de la banda sonora de *Sleepwalkers* está mostrada en la figura D.4 y pertenece a Fantasía.

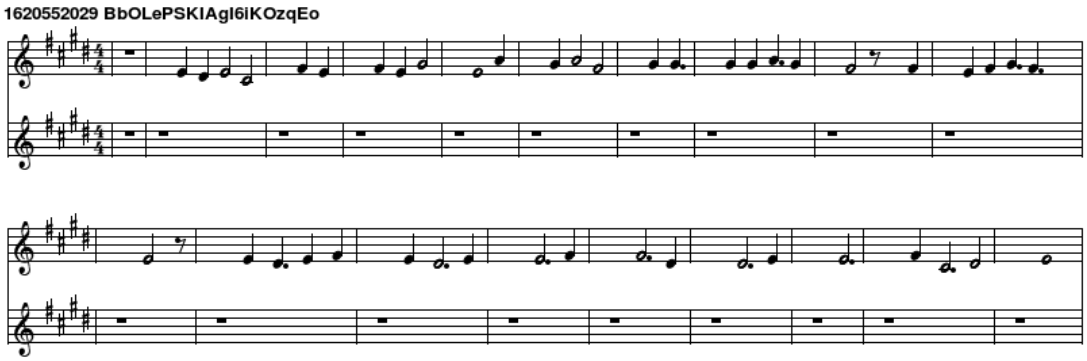


Figura D.4: Composición extraída de Sleepwalkers

Los resultados obtenidos en este apartado sobre diferenciar la composición artificial de la real están en la figura D.5, mientras que los resultados de clasificar cada composición por género cinematográfico están en la figura D.6 y D.7.

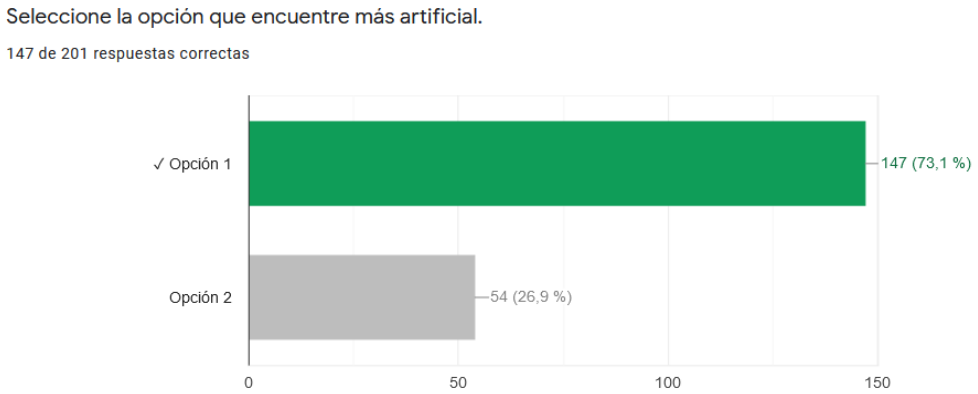


Figura D.5: Resultados 1: Elección de composición artificial

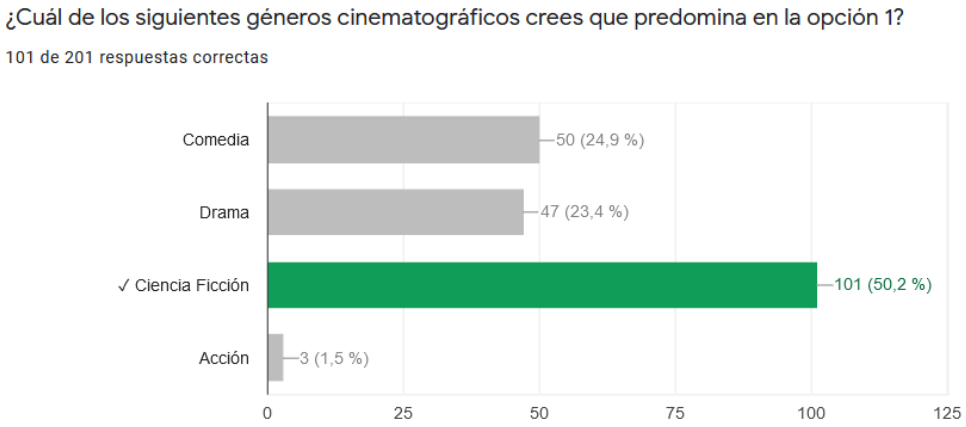


Figura D.6: Resultados 1: Elección género en artificial

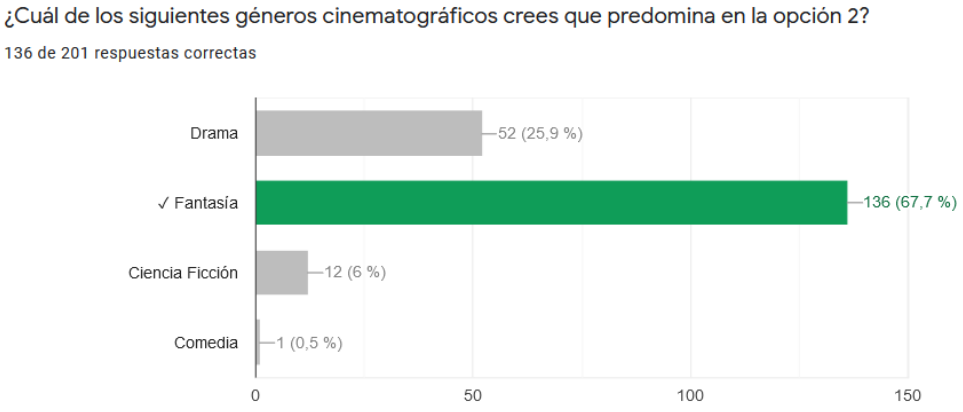


Figura D.7: Resultados 1: Elección género en real

D.3. Apartado 2

En el siguiente enlace se reproducen las dos composiciones de este apartado: `https://youtu.be/ZF71E1eQ-rA`.

La composición extraída de la banda sonora de *Cousins* está mostrada en la figura D.8 y pertenece a Drama.

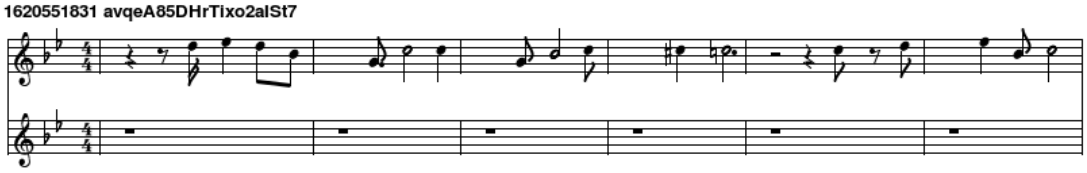


Figura D.8: Composición extraída de Cousins

La composición artificial está mostrada en la figura D.9 y pertenece a Acción.

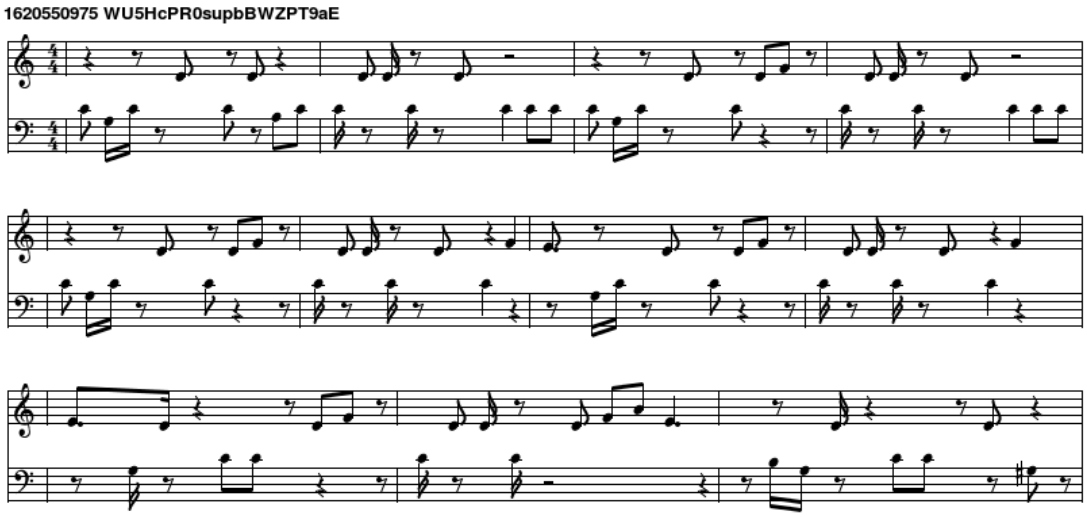


Figura D.9: Segunda composición artificial

Los resultados obtenidos en este apartado sobre diferenciar la composición artificial de la real están en la figura D.10, mientras que los resultados de clasificar cada composición por género cinematográfico están en la figura D.11 y D.12.

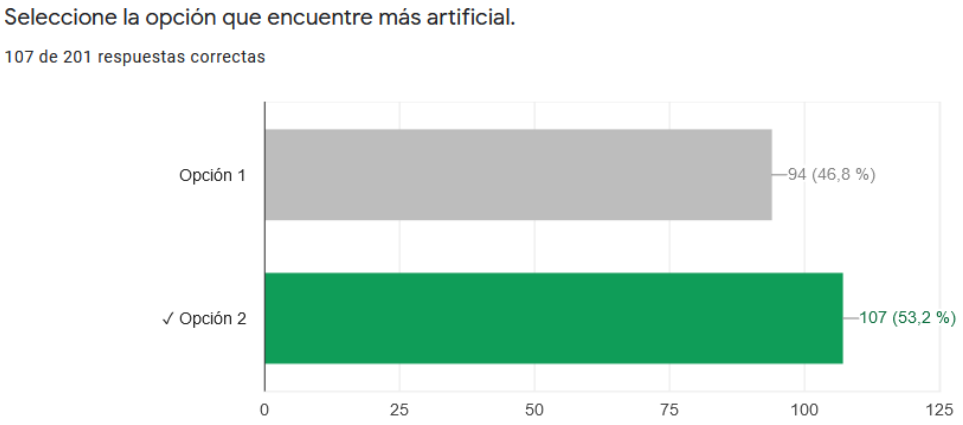


Figura D.10: Resultados 2: Elección de composición artificial

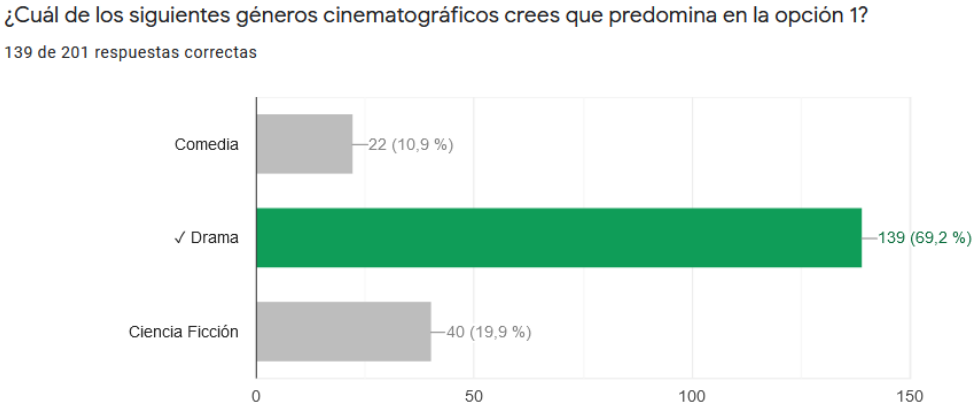


Figura D.11: Resultados 2: Elección género en real

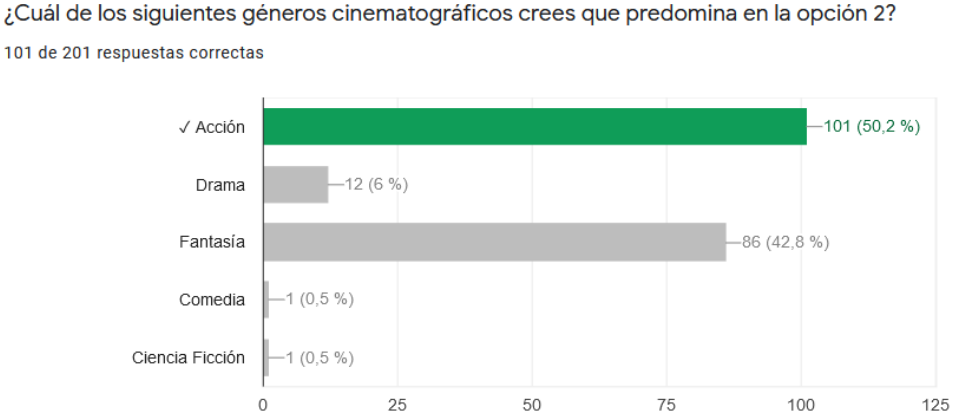


Figura D.12: Resultados 2: Elección género en artificial

D.4. Apartado 3

En el siguiente enlace se reproducen las dos composiciones de este apartado: `https://youtu.be/CXv71oMiWfI`.

La composición extraída de la banda sonora de *Spiderman* está mostrada en la figura D.13 y pertenece a Ciencia Ficción.

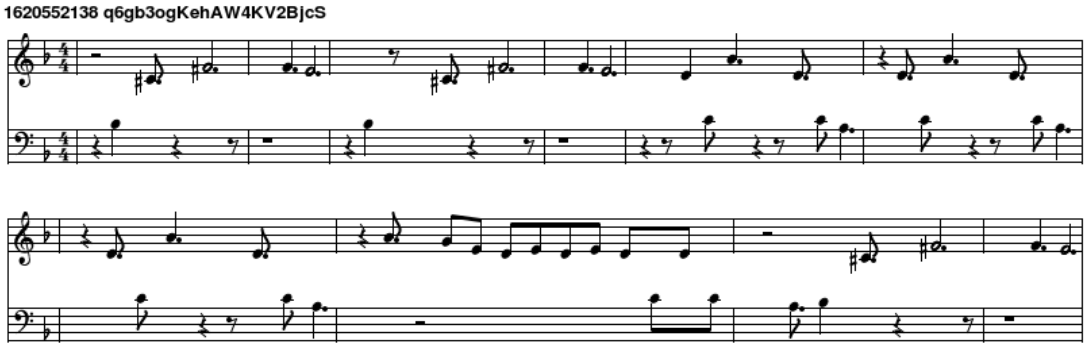


Figura D.13: Composición extraída de Spiderman

La composición artificial está mostrada en la figura D.14 y pertenece a Fantasía.

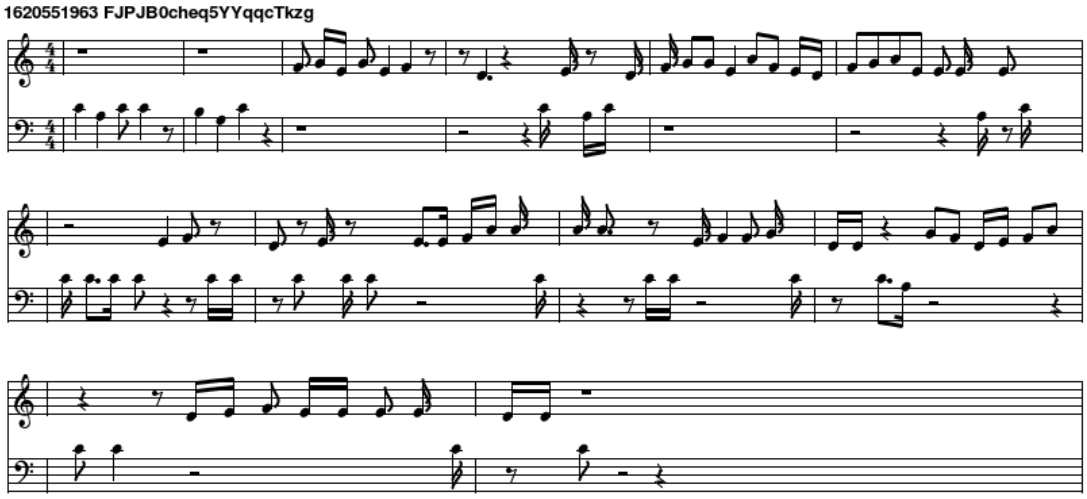


Figura D.14: Tercera composición artificial

Los resultados obtenidos en este apartado sobre diferenciar la composición artificial de la real están en la figura D.15, mientras que los resultados de clasificar cada composición por género cinematográfico están en la figura D.16 y D.17.

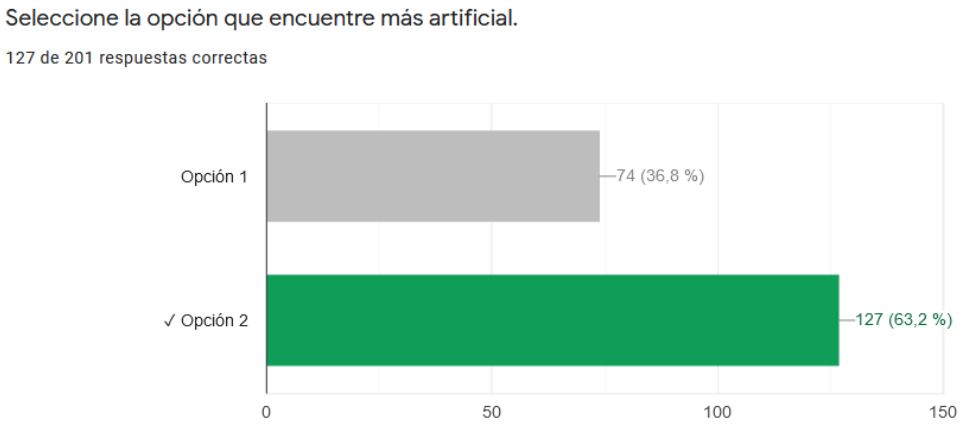


Figura D.15: Resultados 3: Elección de composición artificial

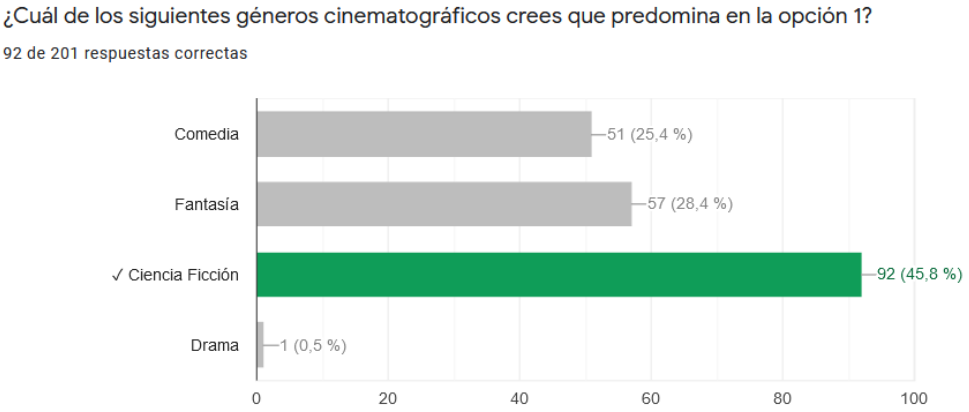


Figura D.16: Resultados 3: Elección género en real

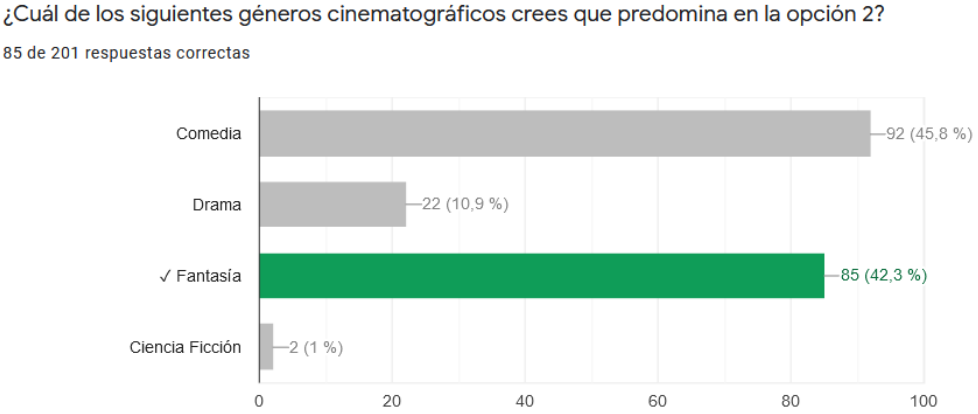


Figura D.17: Resultados 3: Elección género en artificial

D.5. Apartado 4

En el siguiente enlace se reproducen las composiciones de este apartado: `https://youtu.be/M7G4OUkm7Fg`.

La composición artificial está mostrada en la figura D.18 y pertenece a Drama.

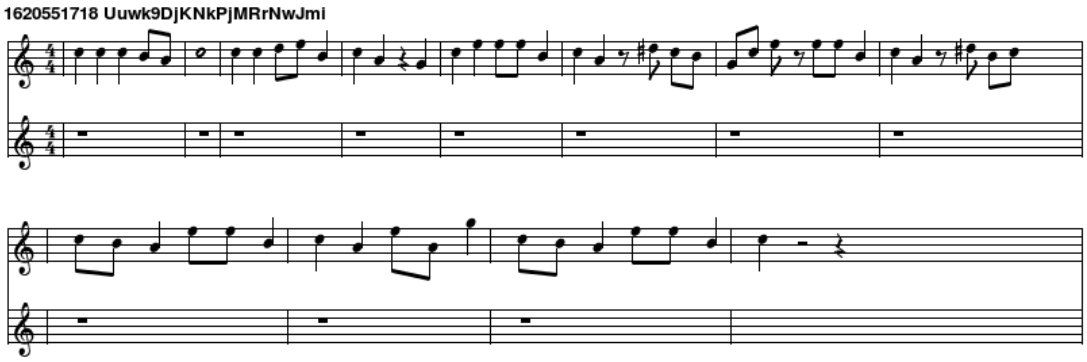


Figura D.18: Cuarta composición artificial

La composición extraída de la banda sonora de *Animal House - Louie, Louie* está mostrada en la figura D.19 y pertenece a Comedia.

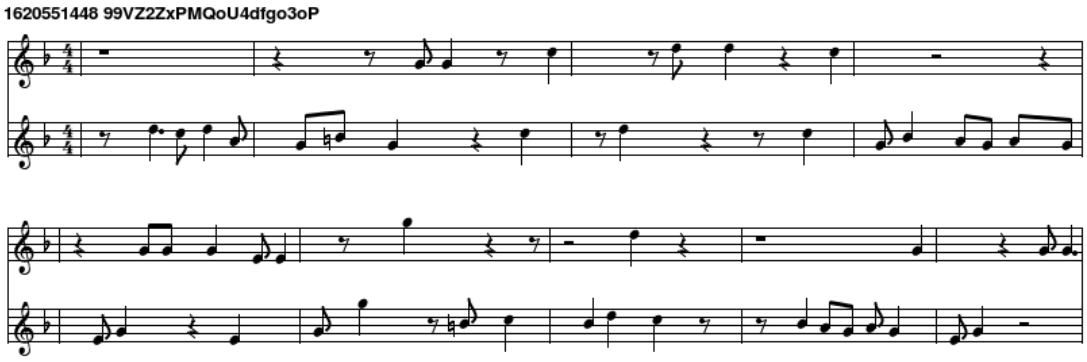


Figura D.19: Composición extraída de Animal House - Louie, Louie

Los resultados obtenidos en este apartado sobre diferenciar la composición artificial de la real están en la figura D.20, mientras que los resultados de clasificar cada composición por género cinematográfico están en la figura D.21 y D.22.

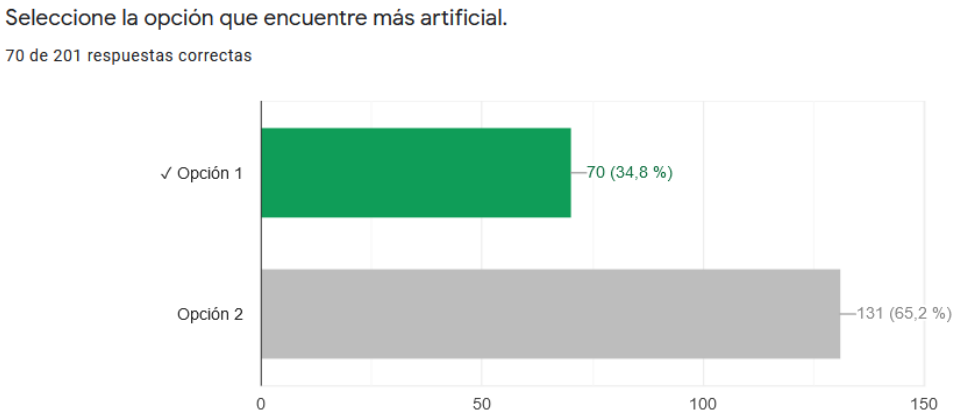


Figura D.20: Resultados 4: Elección de composición artificial

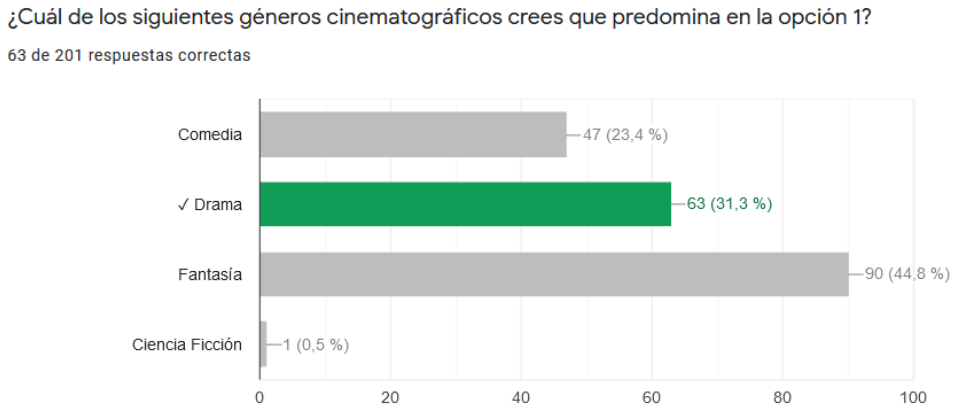


Figura D.21: Resultados 4: Elección género en artificial

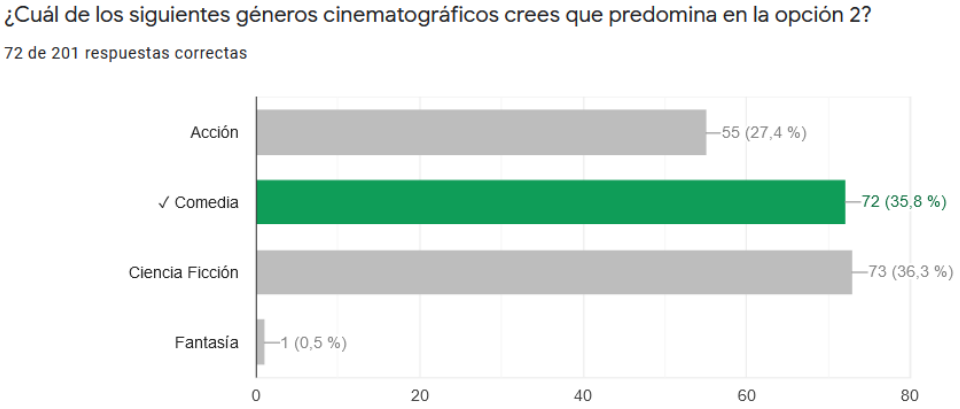


Figura D.22: Resultados 4: Elección género en real

D.6. Apartado 5

En el siguiente enlace se reproducen las composiciones de este último apartado: `https://youtu.be/i-xFJDweRdk`.

La composición artificial está mostrada en la figura D.23 y pertenece a Comedia.

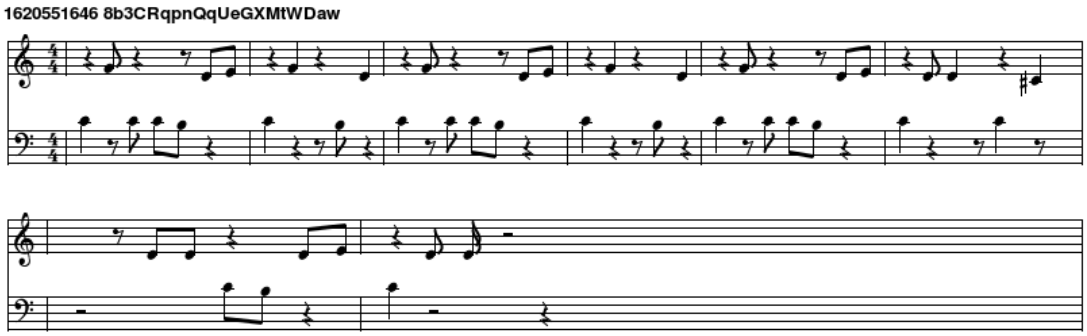


Figura D.23: Quinta composición artificial

La composición extraída de la banda sonora de *Die Hard With A Vengence* está mostrada en la figura D.24 y pertenece a Acción.

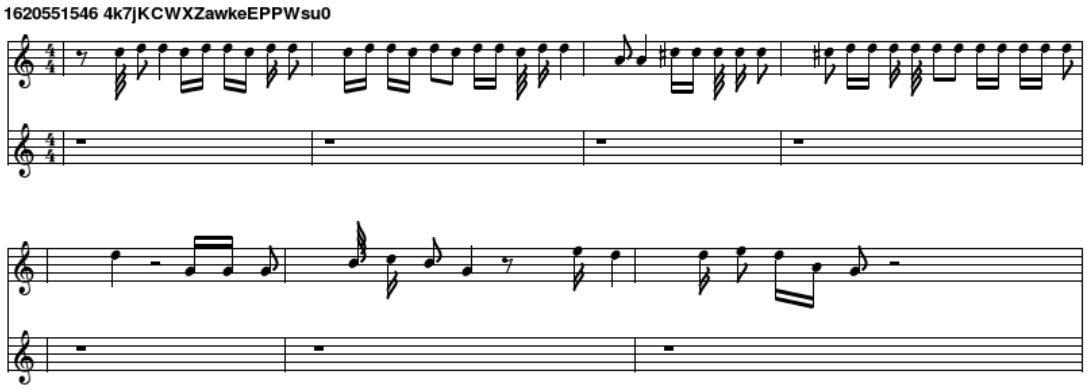


Figura D.24: Composición extraída de Die Hard With A Vengence

Los resultados obtenidos en este apartado sobre diferenciar la composición artificial de la real están en la figura D.25, mientras que los resultados de clasificar cada composición por género cinematográfico están en la figura D.26 y D.27.

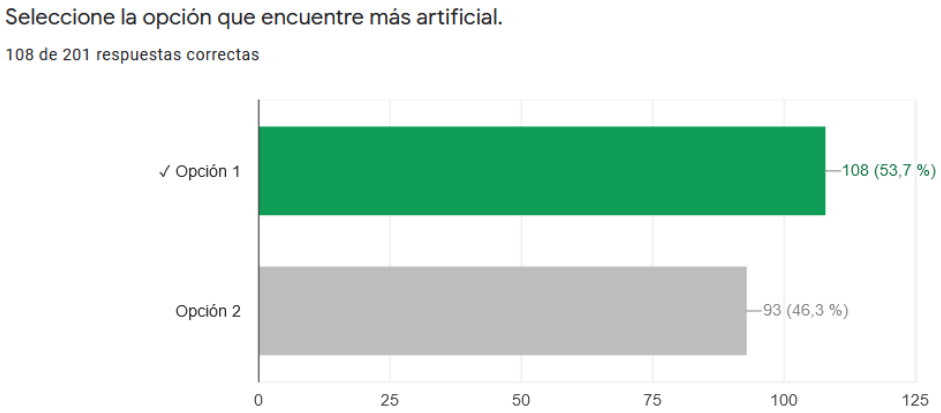


Figura D.25: Resultados 5: Elección de composición artificial

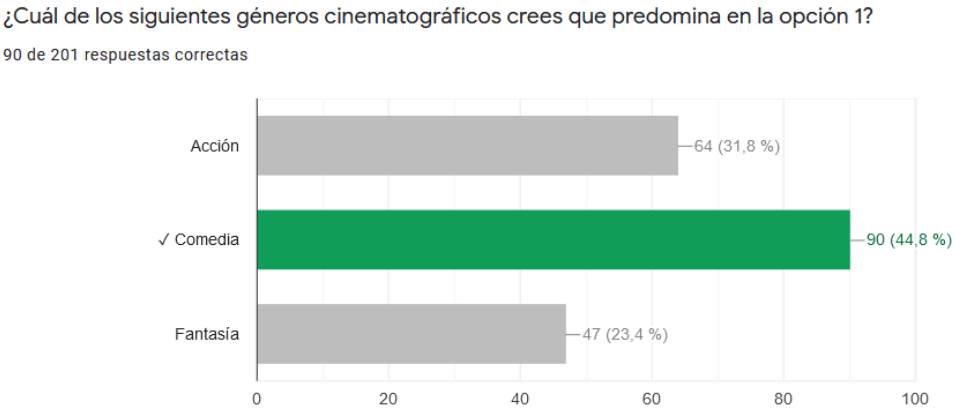


Figura D.26: Resultados 5: Elección género en artificial

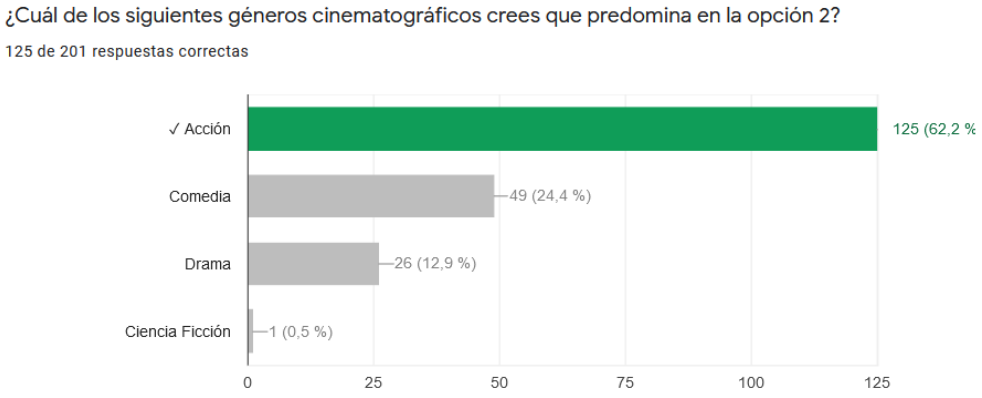


Figura D.27: Resultados 5: Elección género en real